JOINT OFFLOADING DECISION AND RESOURCE ALLOCATION FOR MOBILE CLOUD WITH COMPUTING ACCESS POINT

Meng-Hsi Chen* *Min Dong*† *Ben Liang**

* Dept. of Electrical and Computer Engineering, University of Toronto, Canada

† Dept. of Electrical, Computer and Software Engineering, University of Ontario Institute of Technology, Canada

ABSTRACT

We consider a mobile cloud computing system consisting of multiple users, one computing access point (CAP), and one remote cloud server. The CAP can either process the received tasks from mobile users or offload them to the cloud. We aim to jointly optimize the offloading decisions of all users and the CAP, together with communication and processing resource allocation, to minimize the overall cost of energy, computation, and the maximum delay among all users. It is shown that the problem can be formulated as a non-convex quadratically constrained quadratic program, which is NP-hard in general. We further propose an efficient solution to this problem by semidefinite relaxation and a novel randomization mapping method. Our simulation results show that the proposed algorithm gives nearly optimal performance with only a small number of randomization iterations.

***Index Terms*—** Mobile cloud computing, computing access point, offloading decision, resource allocation, semidefinite relaxation

1. INTRODUCTION

Mobile cloud computing extends the capabilities of mobile devices to improve user experience [1] [2]. Mobile users can offload tasks to the cloud, using abundant cloud resources to help them gather, store, and process data. However, the interaction between mobile devices and the cloud introduces difficult challenges such as the tradeoff between energy savings and computing performance while offloading tasks to the cloud. Besides, the communication delay between mobile users and the cloud is an additional cost that cannot be ignored [3].

An architecture was proposed in [4] to reduce transmission latency by using nearby cloudlets to replace the remote cloud. The scenario of a single user offloading its entire application to the cloud was studied in [5] [6]. The multi-user scenarios were addressed in [7] [8]. Furthermore, in contrast to whole application offloading above, the authors of [9–12] considered application partitioning. In [9], a system named MAUI was proposed to efficiently process the partitioned tasks. Further improvements were proposed by Clonecloud [10] and Thinkair [11]. Dynamic partitioning was considered in [12]. In all cases, the partitioning problem results in integer programming that are difficult to solve.

In our previous work [13], we have studied the impact of a novel Computing Access Point (CAP), which can be a wireless access point or a cellular base station with built-in computation capability, showing substantial system performance improvement for the *single-user* scenario. The CAP can either process the received tasks from the mobile user or offload them to the cloud to provide additional computation capability over traditional mobile cloud computing systems. In this work, we further study the interaction between *multiple users* and the CAP. Both the offloading decision and resource allocation among all users are jointly considered. Different from [13], in this multi-user scenario, we need to allocate communication and computation resources among competing users, with an aim to conserve energy and maintain quality of service for all of them. A new method is required to solve this problem, and an optimal offloading decision must take into consideration the computation and communication energies, computation costs, and communication and processing delays at all local user devices, as well as the resource constraints and capabilities of the CAP and the remote cloud.

We focus on jointly optimizing the offloading decisions as well as the communication and computation resource allocation for multiple mobile users with one CAP and one remote cloud server. We aim to minimize a weighted sum of the costs of energy, computation, and the maximum delay among all users. We show that the resultant mixed integer programming problem can be formulated as a non-convex quadratically constrained quadratic program (QCQP) [14], which is NP-hard in general. To solve this challenging problem, we propose an efficient heuristic algorithm, termed *shareCAP*, based on semidefinite relaxation (SDR) [15] and a novel randomization mapping method. Compared with the optimal offloading policy obtained by exhaustive search, our proposed algorithm gives nearly optimal performance with a small number of randomization iterations. Furthermore, we observe that the addition of a CAP can significantly reduce the energy and computational costs of mobile cloud computing in the multi-user scenario, over traditional systems where only the remote cloud server is available for task offloading.

2. SYSTEM MODEL AND PROBLEM FORMULATION

Consider a cloud access network with N mobile users, one CAP, and one remote cloud server, as shown in Fig. 1. The connections between mobile users and the CAP are wireless, while a wired connection is used between the CAP and the cloud. For the CAP, instead of just serving as a relay to always forward received tasks from users to the cloud, we assume it also has the capability to process user tasks subject to its resource constraint. Each mobile user has one task to be either processed locally or offloaded through a two-phase procedure. In phase one, each mobile user decides whether to offload its task to the CAP, and in phase two, the CAP determines whether to process each received task itself or offload it to the cloud for processing. Since there are multiple tasks offloaded to the CAP and some of them are processed by the CAP, we need to further allocate the communication and computation resources available to the CAP.

Denote the input and output data sizes and the application type of each user i's task by $D_{\text{in}}(i)$, $D_{\text{out}}(i)$, and App(i), respectively, where App(i) refers to the number of processing cycles per input

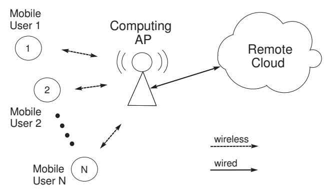

Fig. 1. System model

data in this work. Denote the offloading decisions by

$$x_{l_i} + x_{a_i} + x_{c_i} = 1,\ \ i = 1, \dots, N,$$

where $x_{l_i}, x_{a_i}, x_{c_i} \in \{0, 1\}$ indicate whether user i's task is processed locally, at the CAP, or at the cloud, respectively. Notice that only one of $x_{l_i}, x_{a_i}, x_{c_i}$ for user i could be 1 at the same time.

For the task being locally processed by the mobile user i, denote the corresponding energy consumption for processing it by E_{l_i} and the processing time by T_{l_i}. If the task is offloaded to the CAP, denote the energy consumed for transmitting and receiving by E_{t_i} and E_{r_i} respectively. Furthermore, denote the uplink and downlink transmission times by $T_{t_i} = D_{\text{in}}(i)/c_{u_i}$ and $T_{r_i} = D_{\text{out}}(i)/c_{d_i}$, respectively, where c_{u_i} and c_{d_i} are uplink and downlink bandwidths allocated to user i. Their values depend on the corresponding wireless link capacities, denoted by C_{UL} and C_{DL}, respectively, and the number of other tasks offloaded to the CAP. If this task is processed by the CAP, denote its processing time by $T_{a_i} = D_{\text{in}}(i)\text{App}(i)/f_{a_i}$, where f_{a_i} is the assigned processing rate depending on the CPU rate f_A and the number of tasks being processed at the CAP. If instead the task is further offloaded to the cloud from the CAP, we denote the required transmission time between the CAP and the cloud by T_{ac_i}, and the cloud processing time by T_{c_i}. We assume the transmission between the CAP and the cloud as well as the cloud processing for each offloaded task are separated from the rest and fixed, so that T_{ac_i} and T_{c_i} only depend on each task itself. Finally, denote by C_{a_i} and C_{c_i} the costs of processing user i's task at the CAP and the cloud, respectively.

Since our goal is to reduce the mobile users' energy consumption and maintain the QoS to their tasks, we define the total system cost as the weighted sum of total energy consumption, the costs to offload and process all tasks, and the corresponding maximum transmission and processing delays among all users. We aim to minimize the total system cost by jointly optimizing the task offloading decisions $\mathbf{x}_i = (x_{l_i}, x_{a_i}, x_{c_i})$ as well as the communication and CAP processing resource allocation $\mathbf{r}_i = (c_{u_i},\ c_{d_i},\ f_{a_i})$. The optimization problem is formulated as follows:

$$\min_{\{\mathbf{x}_i\},\{\mathbf{r}_i\}} \Bigg[\sum_{i=1}^{N}(E_{l_i}x_{l_i} + E_{A_i}x_{a_i} + E_{C_i}x_{c_i}) + \max_i\{\rho_i(T_{L_i} + T_{A_i} + T_{C_i})\}\Bigg] \tag{1}$$

$$\text{s.t.}\ \ x_{l_i} + x_{a_i} + x_{c_i} = 1,\ \ i = 1, \dots, N, \tag{2}$$

$$\sum_{i=1}^{N} c_{u_i} \le C_{UL}, \tag{3}$$

$$\sum_{i=1}^{N} c_{d_i} \le C_{DL}, \tag{4}$$

$$\sum_{i=1}^{N} f_{a_i} \le f_A, \tag{5}$$

$$c_{u_i}, c_{d_i}, f_{a_i} \ge 0,\ \ i = 1, \dots, N, \tag{6}$$

$$x_{l_i}, x_{a_i}, x_{c_i} \in \{0, 1\},\ \ i = 1, \dots, N, \tag{7}$$

where $E_{A_i} \triangleq (E_{t_i} + E_{r_i} + \alpha C_{a_i})$ and $E_{C_i} \triangleq (E_{t_i} + E_{r_i} + \beta C_{c_i})$ are the weighted transmission energy and processing cost for task i being offloaded to the CAP or the cloud, respectively, with α and β being their relative weights, $T_{L_i} \triangleq T_{l_i}x_{l_i}$ is the processing delay at the mobile user, $T_{A_i} \triangleq (D_{\text{in}}(i)/c_{u_i} + D_{\text{out}}(i)/c_{d_i} + D_{\text{in}}(i)\text{App}(i)/f_{a_i})x_{a_i}$ and $T_{C_i} \triangleq (D_{\text{in}}(i)/c_{u_i} + D_{\text{out}}(i)/c_{d_i} + T_{ac_i} + T_{c_i})x_{c_i}$ correspond to the transmission and processing delay at the CAP and the cloud, respectively, and ρ_i is the weight on the delay relative to energy consumption. We can adjust ρ_i to place different emphasis on delay and energy consumption. In the above problem formulation, (2) is the offloading placement constraint, (3) and (4) correspond to uplink and downlink bandwidth resource constraints, and (5) is the constraint on the CAP processing resource allocation.

3. SHARECAP OFFLOADING SOLUTION

Given some offloading decisions $\mathbf{x}_i$, the above optimization problem concerns only the resource allocation part and will become

$$\min_{\{\mathbf{r}_i\}} \ \left(\mathbf{E} + \max_i\{\rho_i(T_{L_i} + T_{A_i} + T_{C_i})\}\right) \tag{8}$$
$$\text{s.t.}\ \ (3) - (6),$$

where $\mathbf{E} \triangleq \sum_{i=1}^{N}[E_{l_i}x_{l_i} + E_{A_i}x_{a_i} + E_{C_i}x_{c_i}]$ is a constant depending on $\mathbf{x}_i$. Notice that our original optimization problem (1) is a mixed integer linear programming problem, while the resource allocation problem (8) is convex. In order to find an efficient solution to the original nonconvex problem (1), in the following, we first transform it into a QCQP with a linear objective, and then propose an SDR approach and a novel randomization mapping method to recover the binary offloading decisions. Once we obtain the binary offloading decisions, we can easily solve problem (8) to find the corresponding optimal resource allocation. We name this method the *shareCAP* offloading solution.

3.1. Semidefinite Relaxation

To convert the optimization problem (1) into a QCQP problem, we first replace the integer constraint (7) by

$$x_{k_i}(x_{k_i} - 1) = 0,\ i = 1, \dots, N, \tag{9}$$

for $k = l, a, c$, and introduce additional auxiliary variables t and $\mathbf{d}_i = (D_{u_i}, D_{d_i}, D_{a_i})$. The optimization problem (1) can be rewritten as

$$\min_{\{\mathbf{x}_i\},\{\mathbf{r}_i\},\{\mathbf{d}_i\},t} \ \sum_{i=1}^{N}[E_{l_i}x_{l_i} + E_{A_i}x_{a_i} + E_{C_i}x_{c_i}] + t \tag{10}$$
$$\begin{aligned}\text{s.t.}\ \ & \rho_i(T_{l_i}x_{l_i} + D_{u_i} + D_{d_i} + D_{a_i} \\ & + (T_{ac_i} + T_{c_i})x_{c_i}) \le t,\ \ i = 1, \dots, N, \\ & D_{\text{in}}(i)(x_{a_i} + x_{c_i}) - c_{u_i}D_{u_i} \le 0,\ i = 1, \dots, N, \\ & D_{\text{out}}(i)(x_{a_i} + x_{c_i}) - c_{d_i}D_{d_i} \le 0,\ i = 1, \dots, N, \\ & D_{\text{in}}(i)\text{App}(i)x_{a_i} - f_{a_i}D_{a_i} \le 0,\ i = 1, \dots, N, \\ & (2) - (7).\end{aligned}$$

Define $\mathbf{w} = [\mathbf{x}_1, \dots, \mathbf{x}_N, f_{a_1}, \dots, f_{a_N}, D_{a_1}, \dots, D_{a_N}, c_{u_1}, \dots, c_{u_N}, D_{u_1}, \dots, D_{u_N}, c_{d_1}, \dots, c_{d_N}, D_{d_1}, \dots, D_{d_N}, t]^T$, and $\mathbf{0}$ as the $N \times N$ zero matrix. In addition, define $\mathbf{e}_i$ and $\mathbf{e}'_i$ as the $N \times 1$ and $(9N+1) \times 1$ standard unit vectors with the ith entry being 1, respectively. The optimization problem (1) can now be further transformed into the following QCQP formulation:

$$\begin{aligned}
\min_{\mathbf{w}} \quad & \mathbf{b}_0^T \mathbf{w} & (11)\\
\text{s.t.} \quad & \mathbf{b}_{ci}^T \mathbf{w} \le 0, \;\; i = 1, \dots, N, \\
& \mathbf{w}^T \mathbf{A}_{si} \mathbf{w} + \mathbf{b}_{si}^T \mathbf{w} \le 0, \;\; s = u, d, a; \;\; i = 1, \dots, N, \\
& \mathbf{b}_{Ii}^T \mathbf{w} = 1, \;\; i = 1, \dots, N, \\
& \mathbf{b}_U^T \mathbf{w} \le C_{\text{UL}}, \; \mathbf{b}_D^T \mathbf{w} \le C_{\text{DL}}, \; \mathbf{b}_A^T \mathbf{w} \le f_A, \\
& \mathbf{w}^T \text{diag}(\mathbf{e}'_p) \mathbf{w} - \mathbf{e}'^T_p \mathbf{w} = 0, \;\; p = 1, \dots, 3N, \\
& \mathbf{w} \ge \mathbf{0},
\end{aligned}$$

where

$$\mathbf{A}_{u_i} = \begin{bmatrix} \mathbf{0}_{5N\times 5N} & \mathbf{0}_{5N\times 2N} & \mathbf{0}_{5N\times(2N+1)} \\ \mathbf{0}_{2N\times 5N} & \mathbf{A}'_{u_i} & \mathbf{0}_{2N\times(2N+1)} \\ \mathbf{0}_{(2N+1)\times 5N} & \mathbf{0}_{(2N+1)\times 2N} & \mathbf{0}_{(2N+1)\times(2N+1)} \end{bmatrix},$$

$$\mathbf{A}_{d_i} = \begin{bmatrix} \mathbf{0}_{7N\times 7N} & \mathbf{0}_{7N\times 2N} & \mathbf{0}_{7N\times 1} \\ \mathbf{0}_{2N\times 7N} & \mathbf{A}'_{d_i} & \mathbf{0}_{2N\times 1} \\ \mathbf{0}_{1\times 7N} & \mathbf{0}_{1\times 2N} & 0 \end{bmatrix},$$

$$\mathbf{A}_{a_i} = \begin{bmatrix} \mathbf{0}_{3N\times 3N} & \mathbf{0}_{3N\times 2N} & \mathbf{0}_{3N\times(4N+1)} \\ \mathbf{0}_{2N\times 3N} & \mathbf{A}'_{a_i} & \mathbf{0}_{2N\times(4N+1)} \\ \mathbf{0}_{(4N+1)\times 3N} & \mathbf{0}_{(4N+1)\times 2N} & \mathbf{0}_{(4N+1)\times(4N+1)} \end{bmatrix},$$

$$\mathbf{A}'_{s_i} = -0.5 \begin{bmatrix} \mathbf{0} & \text{diag}(\mathbf{e}_i) \\ \text{diag}(\mathbf{e}_i) & \mathbf{0} \end{bmatrix}, \;\; s = u, d, a,$$

$$\begin{aligned}
&\mathbf{b}_0 = [E_{l_1} \; E_{A_1} \; E_{C_1} \dots E_{l_N} \; E_{A_N} \; E_{C_N} \; \mathbf{0}_{1\times 6N} \; 1]^T, \\
&\mathbf{b}_{ci} = \rho_i [T_{l_i} \mathbf{e}'_{3i-2} + (T_{ac_i} + T_{c_i}) \mathbf{e}'_{3i} \\
&\qquad + \mathbf{e}'_{4N+i} + \mathbf{e}'_{6N+i} + \mathbf{e}'_{8N+i}], \\
&\mathbf{b}_{ui} = \mathbf{D}_{\text{in}}(i)(\mathbf{e}'_{3i-1} + \mathbf{e}'_{3i}), \; \mathbf{b}_{di} = \mathbf{D}_{\text{out}}(i)(\mathbf{e}'_{3i-1} + \mathbf{e}'_{3i}), \\
&\mathbf{b}_{ai} = \mathbf{D}_{\text{in}}(i)\mathbf{App}(i)\mathbf{e}'_{3i-1}, \; \mathbf{b}_{Ii} = \mathbf{e}'_{3i-2} + \mathbf{e}'_{3i-1} + \mathbf{e}'_{3i}, \\
&\mathbf{b}_U = [\mathbf{0}_{1\times 5N} \; \mathbf{1}_{1\times N} \; \mathbf{0}_{1\times(3N+1)}]^T, \\
&\mathbf{b}_D = [\mathbf{0}_{1\times 7N} \; \mathbf{1}_{1\times N} \; \mathbf{0}_{1\times(N+1)}]^T, \\
&\mathbf{b}_A = [\mathbf{0}_{1\times 3N} \; \mathbf{1}_{1\times N} \; \mathbf{0}_{1\times(5N+1)}]^T.
\end{aligned}$$

Comparing the optimization problems (10) and (11), all constraints have one-to-one correspondence. Note that the optimization problem (11) is a non-convex QCQP problem, which is NP-hard in general. To solve it, we apply the SDR approach to relax it into a semidefinite programming (SDP) problem. Define $\mathbf{X} = [\mathbf{w}^T \; 1]^T [\mathbf{w}^T \; 1]$. By dropping the rank constraint $\text{rank}(\mathbf{X}) = 1$, we have the following SDP problem:

$$\begin{aligned}
\min_{\mathbf{X}} \quad & \text{Tr}(\mathbf{G}_0 \mathbf{X}) & (12)\\
\text{s.t.} \quad & \text{Tr}(\mathbf{G}_{r_i} \mathbf{X}) \le 0, \;\; r = c, u, d, a; \;\; i = 1, \dots, N, \\
& \text{Tr}(\mathbf{G}_{I_i} \mathbf{X}) = 1, \;\; i = 1, \dots, N, \\
& \text{Tr}(\mathbf{G}_U \mathbf{X}) \le C_{\text{UL}}, \; \text{Tr}(\mathbf{G}_D \mathbf{X}) \le C_{\text{DL}}, \; \text{Tr}(\mathbf{G}_A \mathbf{X}) \le f_A, \\
& \text{Tr}(\mathbf{G}_p \mathbf{X}) = 0, \;\; p = 1, \dots, 3N, \\
& \mathbf{X}(9N+2, 9N+2) = 1, \;\; \mathbf{X} \succeq \mathbf{0},
\end{aligned}$$

where

$$\mathbf{G}_0 = \begin{bmatrix} \mathbf{0}_{(9N+1)\times(9N+1)} & \frac{1}{2}\mathbf{b}_0 \\ \frac{1}{2}\mathbf{b}_0^T & 0 \end{bmatrix},$$

Algorithm 1 ShareCAP Offloading Algorithm

```
1: Obtain optimal solution $\mathbf{X}^*$ of the SDP problem (12). Extract
   the upper-left $3N \times 3N$ sub-matrix $\mathbf{X}'^*$ from $\mathbf{X}^*$. Set the num-
   ber of randomization trials as $L$.
2: Record the values of diagonal terms in $\mathbf{X}'^*$ as $\mathbf{p} =
   [p_{l_1} p_{a_1} p_{c_1} \dots p_{l_N} p_{a_N} p_{c_N}]^T$.
3: for $l = 1, ..., L$ do
4:     $\mathbf{v}^{(l)} = [\mathbf{u}_1 \dots \mathbf{u}_N]^T$ with random vectors generated in (13);
5:     Solve resource allocation problem (8) given offloading
       decision $\mathbf{v}^{(l)}$;
6: end for
7: Choose among $\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(L)}$ the one that yields the minimum
   objective value of (8); Set it as $\mathbf{v}^o$.
8: Output: the proposed offloading solution $\mathbf{v}^o$ and the correspond-
   ing resource allocation.
```

$$\mathbf{G}_{l_i} = \begin{bmatrix} \mathbf{0}_{(9N+1)\times(9N+1)} & \frac{1}{2}\mathbf{b}_{l_i} \\ \frac{1}{2}\mathbf{b}_{l_i}^T & 0 \end{bmatrix}, \;\; l = c, I; \;\; i = 1, \dots, N,$$

$$\mathbf{G}_{s_i} = \begin{bmatrix} \mathbf{A}_{si} & \frac{1}{2}\mathbf{b}_{s_i} \\ \frac{1}{2}\mathbf{b}_{s_i}^T & 0 \end{bmatrix}, \;\; s = u, d, a; \;\; i = 1, \dots, N,$$

$$\mathbf{G}_S = \begin{bmatrix} \mathbf{0}_{(9N+1)\times(9N+1)} & \frac{1}{2}\mathbf{b}_S \\ \frac{1}{2}\mathbf{b}_S^T & 0 \end{bmatrix}, \;\; S = U, D, A,$$

$$\mathbf{G}_p = \begin{bmatrix} \text{diag}(\mathbf{e}'_p) & -\frac{1}{2}\mathbf{e}'_p \\ -\frac{1}{2}\mathbf{e}'^T_p & 0 \end{bmatrix}, \;\; p = 1, \dots, 3N.$$

The above problem can be solved efficiently in polynomial time using standard SDP software, such as SeDuMi [16].

Denote $\mathbf{X}^*$ as the optimal solution of the SDP problem (12). We need to recover a rank-1 solution of the original problem (1) from $\mathbf{X}^*$. In the following, we propose a randomization method to obtain our binary offloading decisions.

3.2. Binary Offloading Decisions via Randomization

A common approach [15] to obtain an integer solution from the relaxed SDP problem is to randomly generate vectors from the Gaussian distribution with zero mean and covariance $\mathbf{X}^*$ for L times, and then map them to the integer set $\{0, 1\}^{3N}$ by using the sign of each element in these vectors. Among the generated vectors, the one that yields the best objective value of the original problem will be chosen as the desired solution. However, the above randomization procedure may not be feasible when those generated vectors cannot satisfy the placement constraint (2). Instead, we propose the following improved method.

Define $\mathbf{v} = [x_{l_1}, x_{a_1}, x_{c_1}, \dots, x_{l_N}, x_{a_N}, x_{c_N}]^T$ as the offloading solution. Notice that, first, only the upper-left $3N \times 3N$ submatrix of $\mathbf{X}^*$, denote by $\mathbf{X}'^*$, is needed to recover the solution $\mathbf{v}$. Second, each diagonal term in $\mathbf{X}'^*$ is always between 0 and 1, denoted by $\mathbf{p} = [p_{l_1}, p_{a_1}, p_{c_1}, \dots, p_{l_N}, p_{a_N}, p_{c_N}]^T$, corresponding to the probability that each element in $\mathbf{v}$ is 1. To satisfy the placement constraint (2), we define $U_{l_i} = p_{l_i}(1 - p_{a_i})(1 - p_{c_i})$, $U_{a_i} = (1 - p_{l_i}) p_{a_i} (1 - p_{c_i})$, $U_{c_i} = (1 - p_{l_i})(1 - p_{a_i}) p_{c_i}$, and randomly select vectors $\mathbf{u}_i$, which represent the location that user i's task will be processed, as follows:

$$\mathbf{u}_i = \begin{cases} (1, 0, 0), & \text{with probability } P_{l_i} \text{ (local processing)}, \\ (0, 1, 0), & \text{with probability } P_{a_i} \text{ (CAP processing)}, \\ (0, 0, 1), & \text{with probability } P_{c_i} \text{ (cloud processing)}, \end{cases} \quad (13)$$

where $P_{l_i} = U_{l_i}/(U_{l_i} + U_{a_i} + U_{c_i})$, $P_{a_i} = U_{a_i}/(U_{l_i} + U_{a_i} + U_{c_i})$, $P_{c_i} = U_{c_i}/(U_{l_i} + U_{a_i} + U_{c_i})$, and $P_{l_i} + P_{a_i} + P_{c_i} = 1$. We gen-

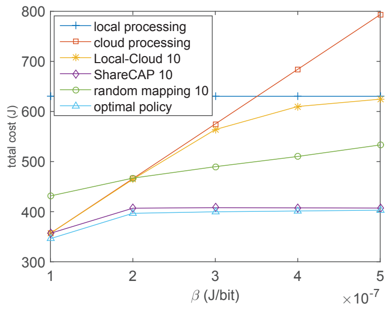

Fig. 2. The total cost under different policies vs. β (J/bit).

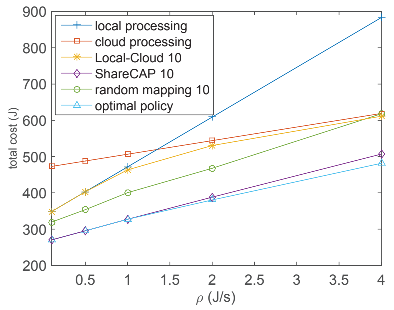

Fig. 3. The total cost under different policies versus ρ (J/s).

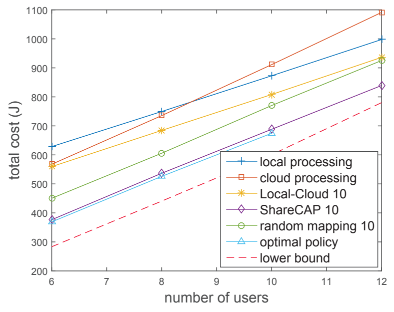

Fig. 4. The total cost under different policies versus number of users.

erate L i.i.d. feasible solutions $\mathbf{v}^{(l)} = [\mathbf{u}_1 \dots \mathbf{u}_N]^T$ using the above procedure, for $l = 1, ..., L$, and solve the corresponding resource allocation problem (8) for each. We then choose the one among these feasible solutions that gives the minimum objective value of the optimization problem (1) to obtain the offloading solution and corresponding resource allocation.

The details of the overall *shareCAP* offloading and resource allocation algorithm are given in Algorithm 1. Notice that the SDP problem (12) can be solved within precision ϵ by the interior point method in $O(\sqrt{N}\log(1/\epsilon))$ iterations, where the amount of work per iteration is $O(N^6)$ [17], while there are 3^N choices in exhaustive search to find the optimal offloading decision. In addition, we observe from simulation results that a small number of randomization trials (e.g., $L = 10$) is enough to give system performance very close to the optimal one.

4. SIMULATION RESULTS

In this section, we provide computer simulation to study the performance of our proposed *shareCAP* offloading solution under different parameter settings. In the following, the default parameter values are described, unless otherwise indicated later. We adopt the mobile device characteristics from [18], which is based on Nokia N900, and set the number of users as $N = 6$. According to Tables 1 and 3 in [18], the mobile device has CPU rate 500×10^6 cycles/s and processing energy consumption $\frac{1}{730\times10^6}$ J/cycle, and the local computation time 4.75×10^{-7} s/bit and local processing energy consumption 3.25×10^{-7} J/bit are calculated when the x264 CBR encode application (1900 cycles/byte) is considered as App(i) in our simulations. The input data size $D_{\text{in}}(i)$ of each task is assumed to be uniformly distributed from 1 to 40MB, and the output data size $D_{\text{out}}(i) = D_{\text{in}}(i)/10$.

In addition, both uplink and downlink transmission capacities are 72.2 Mbps (e.g., IEEE 802.11n) between the mobile user and the CAP, and the transmission and receiving energy consumptions of the mobile user are both 1.42×10^{-7} J/bit as indicated in Table 2 in [18]. The CPU rates of the CAP and each sever at the remote cloud are 5×10^9 cycle/s and 10×10^9 cycle/s, respectively. When tasks are offloaded to the cloud, the transmission rate R_{ac} is 15 Mpbs. Also, we set the values of cost C_{a_i} and C_{c_i} to be the same as that of the input data size $D_{\text{in}}(i)$, and $\alpha = 1\times10^{-7}$ J/bit and $\beta = 3\times10^{-7}$ J/bit. We further set $\rho_i = 2$ J/s as the weight of the delay to process each user's task.

For comparison, we also consider the following methods: 1) the *local processing only* method where all tasks are processed by mobile users, 2) the *cloud processing only* method where all tasks are offloaded to the cloud, 3) the *local-cloud offloading* method where the same approximation procedure as the *shareCAP* method is applied except that there is no CAP, 4) the *random mapping* method where each task is processed at different locations with equal probability, 5) the *optimal policy* where the optimal value is obtained by exhaustive search which is possible only when the number of users is small, and 6) the *lower bound* of the minimum cost, which is the optimal objective value of the SDR problem (12), which is used when the number of users is larger.

In the following figures, we use "*shareCAP* 10," "*local-cloud* 10," and "*random mapping* 10" to indicate that $L = 10$ for the randomization trials in these methods. Our simulation shows that it is sufficient for *shareCAP* to provide near-optimal performance, despite the much larger 3^N decision space of the optimization problem. Finally, all simulation results are obtained by averaging over 100 realizations of the input and output data sizes of each task.

In Figs. 2, we show the system cost vs. the weights β on the cloud processing cost. When β becomes large, all tasks are more likely to be processed by either the mobile user or the CAP. The *local-cloud* method in this case converges to the local processing method. In Fig. 3, we study the system cost under various values of weight ρ on the processing delay. We observe that with the help of the CAP, *shareCAP* outperforms all other methods and is nearly optimal.

Finally, we examine the impact of the limited processing capability at the CAP on the performance of *shareCAP*. Fig. 4 plots the total system cost vs. the number of users N. We are not able to obtain an optimal solution for $N > 10$ due the complexity of exhaustive search. However, we see that *shareCAP* is close to the SDR lower bound, indicating that it is nearly optimal for all N values.

5. CONCLUSION

We consider mobile cloud computing system consisting of multiple users, one CAP, and one remote cloud server. We have developed a new method toward minimizing the weighted total cost of energy, computation, and the maximum delay among all users through joint tasks offloading and resource allocation. Although the optimization problem is non-convex, we propose an efficient heuristic algorithm using SDR and a new randomization mapping approach. Simulation results suggest that the proposed algorithm gives nearly optimal performance, and the resultant efficient utilization of a CAP can bring substantial cost benefit.

6. REFERENCES

[1] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mob. Netw. Appl.*, vol. 18, pp. 129–140, 2013.

[2] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, pp. 84 – 106, 2013.

[3] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, pp. 51–56, 2010.

[4] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, pp. 14–23, 2009.

[5] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Computation offloading for mobile cloud computing based on wide cross-layer optimization," in *Proc. Future Network and Mobile Summit (FutureNetworkSummit)*, 2013, pp. 1–10.

[6] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, pp. 4569–4581, 2013.

[7] S. Ren and M. van der Schaar, "Efficient resource provisioning and rate selection for stream mining in a community cloud," *IEEE Transactions on Multimedia*, vol. 15, pp. 723–734, 2013.

[8] O. Munoz, A. Pascual Iserte, J. Vidal, and M. Molina, "Energy-latency trade-off for multiuser wireless computation offloading," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC) Workshops*, 2014, pp. 29–33.

[9] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: Making smartphones last longer with code offload," in *Proc. ACM International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2010, pp. 49–62.

[10] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proc. ACM Conference on Computer Systems (EuroSys)*, 2011, pp. 301–314.

[11] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proc. IEEE International Conference on Computer Communications (INFOCOM)*, 2012, pp. 945–953.

[12] J. Niu, W. Song, L. Shu, and M. Atiquzzaman, "Bandwidth-adaptive application partitioning for execution time and energy optimization," in *Proc. IEEE International Conference on Communications (ICC)*, 2013, pp. 3660–3665.

[13] M.-H. Chen, B. Liang, and M. Dong, "A semidefinite relaxation approach to mobile cloud offloading with computing access point," in *Proc. IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2015.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[15] Z.-Q. Luo, W.-K. Ma, A.-C. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Processing Magazine*, vol. 27, pp. 20–34, 2010.

[16] M. Grant, S. Boyd, and Y. Ye, "CVX: Matlab software for disciplined convex programming," 2009. [Online]. Available: http://cvxr.com/cvx/

[17] Y. Nesterov, A. Nemirovskii, and Y. Ye, *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.

[18] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. the 2nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud)*, 2010.