Mobile Edge Offloading using Markov Decision Processes

Khalid R. Alasmari[1], Robert C. Green II[2] and Mansoor Alam[1]

[1] EECS Department, The University of Toledo, Toledo,OH 43606
[2] Department of Computer Science, Bowling Green State University, Bowling Green, OH 43403

Abstract. Considering where to process data and perform computation is becoming a more difficult problem as Mobile Edge Computing (MEC) and Mobile Cloud Computing (MCC) continue to evolve. In order to balance constraints and objectives regarding items like computation time and energy consumption, computation and data should be automatically shifted between mobile devices, the edge, and the cloud. To address this issue, this study proposes a Markov Decision Process (MDP) based methodology to intelligently make such choices while optimizing multiple objectives. Results demonstrate an 17.47% or greater increase in performance.

1 Introduction

Overall, the number of smart phone users is rapidly increasing. According to a Gartner press release from August 22, 2017 [1], Global sales of smart phones to end users totaled 366.2 million units in the second quarter of 2017, a 6.7 percent increase over the second quarter of 2016. These numbers demonstrate the higher demand for smart phones and their level of integration into everyday life. However, smart phones still face the challenge of performing complex multimedia operations such as image and video processing, object or face recognition, and augmented reality applications [2]. As many of these operations can be computationally intense, maintaining battery life while addressing a consumer's need is becoming a bigger challenge.

In [3], Running tasks on mobile device will consume a large amount of energy and bandwidth. Therefore, many researchers have proposed offloading mechanism to reduce energy consumption on mobile devices by moving all or some of the computation to the cloud [2] [4]. Some of mobile applications, such as perception and multimedia applications, the network latency of the cloud might face a difficulty to achieve the desired performance [5]. Thus, mobiles devices may prefer to access edge servers that have a lower latency for computation offloading.

This study differs from previous studies by adding an edge device between the mobile device and the cloud servers to perform data processing at the edge of the

* Published version available at http://doi.org/10.1007/978-3-319-94340-4_6

network rather than sending them towards the cloud. This reduces end-to-end delay, energy consumption and lower network congestion. Fig. 1 illustrates an architecture of our proposed system model for multisite offloading that include a mobile device, cloud server and edge server. In this study, a methodology is proposed that shows the offload tasks to edge servers is the effective technique to save mobile device energy and reduce time delay. Therfore, we investigate the collaborative application execution between the mobile device, the edge servers and the cloud servers to conserve the energy consumption on the the mobile device by offloading technique.

The rest of the paper is organized as follows. Sect. 2 provides the related work. Then in Sect. 3, a MDP methodology and formulation. Sect. 4 provides a Numerical simulation and evaluation. Finally, Sect. 5 concludes this paper.

2 related work

There are a variety of previous works that exist in the area of multi-site offloading policies using markov decision processes (MDPs) for mobile cloud computing. Terefe et al. [3] proposed a multisite offloading policy (MDP) for mobile devices in order to minimize energy consumption. The authors adopted a discrete time Markov chain (DTMC) to model a fading channel and applied a MDP framework to formulate the energy and time consumption of the multi-site offloading decisions problem. The authors proposed the Energy-efficient Multisite Offloading Policy (EMOP) algorithm with the value iteration algorithm (VIA) to determine the optimal policy for a Markov chain model. However, thier work considers multiple cloud serves. The result shows that the EMOP algorithm is an efficient multisite computation offloading approach for mobile devices with respect to both energy consumption and execution time. Our approch considers edge computing to develop more efficient offloading solution in energy and time.

Nasseri et al. [6] proposed a methodology that allowed various computation tasks to be offloaded to mobile devices that belong to users considering battery life, response delay, and power consumption. The authors adopted MDP optimal policies and lookup tables for mobile cloud computing in order to guide mobile devices in accepting or rejecting requests based on rewards. Result showed higher rewards with a combination of a smaller delay in responding to a request and reduced power consumption. However, sending the mobile phones data to the lookup tables to update the battery level, signal strength and the distance to helper consumes more energy. In this paper, we consider offloading sites have own database server that could do the computational more easier and without spending time and energy to retrieve the information from another location.

Zhang et al., in [4], proposed a framework solution for energy-optimal mobile cloud computing under stochastic wireless channel. The authors adapted a dynamic configuring technique to the clock frequency of the chip in order to minimize the computation energy in mobile device. The authors also developed a formulation that leads to an optimal data transmission schedule across the stochastic wireless channel to minimize the transmission energy in cloud space.

The authors employ the two-state Markov model known as the Gilbert-Elliot channel. The result suggest offloading mobile applications to the cloud in order to save a significant amount of energy in some application. Our approch considers markov decision processes (MDPs) with three channel states (mobile, edge and cloud).

3 MDP methodology and formulation

MDPs are used to help to make decisions in a stochastic environment. A MDP is a discrete time stochastic control process. It is defined by a state space for the system, an action space, a stochastic transition to determine how the system will move to next state, and a reward function which determines the immediate consequence for the agent's choice of action a while in state s. Hence, a Markov decision process can be defined by the 4-tuple (S, A, P, R) with the following meaning:

– S is a finite set of states,
– A is a finite set of actions,
– $P(s, s', a)$ is the probability that action a in state s at time t will lead to state s' at time $t + 1$,
– $R(s, s', a)$ is the immediate reward received after transition to state s' from state s with action a.

3.1 MDP formulation

This section introduces the formulation of our Multisite offloading in mobile edge computing adupting the MDP methodology including appropriate policy constructs (i.e. state space, decision epochs, actions, transition probabilities, policy, and reward function). The system architecture is shown in Fig. 1. Algorithm 2 shows the Energy-efficient multisite offloading policy algorithm.

State Space The state space, S is defined as $S = \{1, 2, 3\}$ where 1 denotes the mobile space, 2 denotes the Edge site 1 and 3 denotes the Cloud.

Decision Epochs and Actions The decision epoch is represented as $T = \{0, 1, 2, \ldots, n, n+1\}$ where decision epoch $t \in T$ indicates that component t has already executed.

Transition probabilities The transition probabilities play the role of the next-state function in a problem-solving search. Accordingly, for each state $s(t)$ and action $a(t)$, the probability that the next state will be $s(t + 1)$.

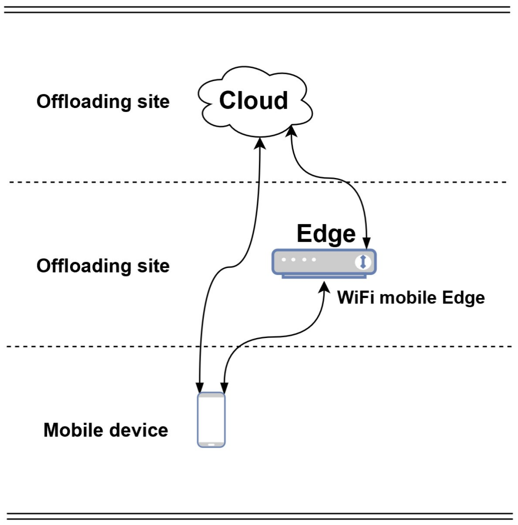

Fig. 1. System model for the multi-site offloading formulation.

Reward Function The reward function in this study considers two components, Energy consumption and computation time, resulting in two objective functions, $R_e(s,a)$ and $R_t(s,a)$. Based on the reward function R and the transition function T_p, a transition is made to state s' with probability T_p and a reward $R(s,s',a)$ is received [7].

Value iteration algorithm The VIA in Algorithm 1 will yield an approximation to the optimal value function and is used in this study.

Algorithm 1 Value Iteration Algorithm (VIA)

```
for all $s \in S$ do $V \leftarrow 0$
end for
repeat $\Delta \leftarrow 0$
    for all $s \in S$ do
$v \leftarrow V(s)$
$V(s) \leftarrow max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$
$\Delta \leftarrow max\,(\Delta, |v - V(s)|)]$

    end for
until $\Delta \leq \theta$( a small positive number )
$\pi(s) \leftarrow argmax_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$
```

4 Numerical simulation and evaluation

In this section, the performance of the proposed MDP-based methodology in terms of energy consumption and computation time is analyzed.

4.1 Simulation setup

This study considers a scenario with two offloading sites ($i.e., K = 2$). Site 1 simulates an edge server and Site 2 simulates a cloud server that also contains a database server as shown in Fig. 1. The mobile application consists of n components in a linear topology, where each component could migrate to one of the two offloading sites or remain on the mobile device in any given step. It is assumed that the two offloading sites have different computational capacity and network bandwidth [3].

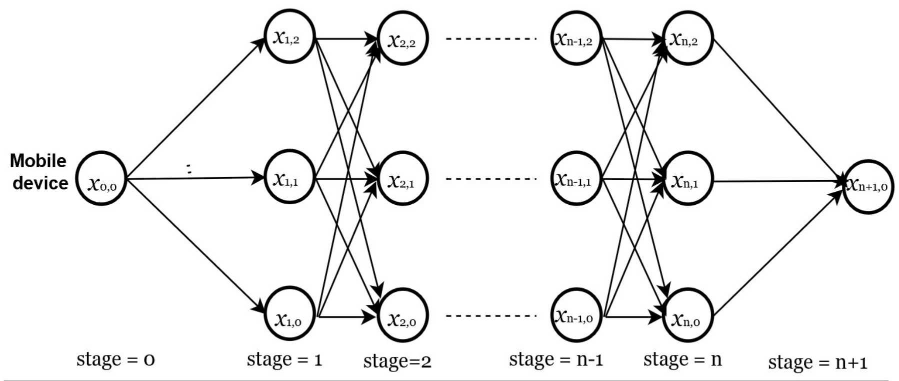

Fig. 2. State transitions in the MDP model.

In the proposed model, the decision epoch is represented as $T = \{0, 1, 2, \ldots, n, n+1\}$, where decision epoch $t \in T$ indicates that component t has already executed. The decision maker chooses either the action taken where $a = 1$ (good) at time t that causes the state to transition to a new state at time $t+1$ from the current state, or the action is not taken where $a = 0$(bad) [3]. In that event, at the beginning of every stage, a decision-maker observes the current state, and chooses an action: migrate execution or continue execution, and receives a reward depending on the current state [8].

The system state at decision epoch t and $i \in [0, k]$ denotes the location of the executed component t. It is assumed that the executions start at the mobile device and the initial channel state is observed to be good, which means computation may be offloaded to one of the two sites. There are $n + 2$ stages of execution considered, as shown in Fig. 2. Stage 0, at decision epoch 0, and stage $n+1$ represent the initiation and termination of application execution, respectively. In each stage, the system state is defined as $x_i = (t_i, \gamma_i)$ where t_i is the location of the executed component t, and γ_i is the channel state of the next time slot between the mobile and offloading sites(i.e., ether good or bad).

Since the application execution starts and ends on the mobile device, it also holds that $t_0 = 0$, and $t_{n+1} = 0$.

The execution sites are defined as Q $= \{q_0, q_1, q_2\}$, where q_0 represents the mobile device, q_1 denotes the offloading site 1 (edge server) and q_2 denotes the offloading site 2 (cloud server). Energy cost is defined as $E_v = \{e_{v_0}, e_{v_1}, e_{v_2}\}$, where e_{v_0}, e_{v_1} and e_{v_2} denotes the energy cost of component v that executed on offloading site q_0, q_1 and q_2,respectively. Time cost is defined as $T_v = \{t_{v_0}, t_{v_1}, t_{v_2}\}$, where t_{v_0}, t_{v_1} and t_{v_2} denotes the time cost to execute component v on each of the offloading sites q_0, q_1 and q_2, respectively[3].

f_0, f_1 and f_2 are defined as the CPU clock speeds (cycles/second) of mobile device q_0, offloading site 1 (q_1), and offloading site 2 (q_2). The total CPU cycles needed by the instructions of component v is W_v. $t^c_{v_i}$ denotes the computational time of executing component v on site q_i and is given by:

$$t^c_{v_i} = \frac{w_v}{f_i} \forall v \in V \text{ and } \forall i \in [0, k] \tag{1}$$

Data sent and received by component v as denoted as d_{v_s} and d_{v_r}, respectively. Since the database is located at the cloud site, r_0 and r_1 are defined as the data rate between site q_0 and site q_1 and the database server [3]. Also, $t^s_{v_i}$ and $t^r_{v_i}$ are defined as the communication time spent for sending and receiving data from the database by component v on site q_i, given by (2) and (3).

$$t^s_{v_i} = \frac{d_{v_s}}{r_i}, \forall v \in V \text{ and } \forall i \in [0, k] \tag{2}$$

$$t^r_{v_i} = \frac{d_{v_r}}{r_i}, \forall v \in V \text{ and } \forall i \in [0, k] \tag{3}$$

t_{v_i} is defined as the total time cost of component v on site q_i and is given by

$$t_{v_i} = t^c_{v_i} + t^s_{v_i} + t^r_{v_i} \tag{4}$$

It is assumed that the energy consumption E_v is calculated as the amount of energy a mobile device spends while executing the component or waiting for the component to be executed on offloading sites [3]. Energy cost of a component is then defined by e_{v_i} in (5) where p_c is the mobile power consumption when computing, p_s is the mobile power consumption when sending data, p_r is the mobile power consumption when receiving data, and p_{idle} is the Mobile power consumption at idle [3].

$$e_{v_i} = \begin{cases} t^c_{v_i} \times p_c + t^s_{v_i} \times p_s + t^r_{v_i} \times p_r, \\ t_{v_i} \times p_{idle} \end{cases} \tag{5}$$

The communication energy cost between two edges is denoted as $e_{u,v} = \{e_{u_0v_0}, e_{u_0v_1}, \ldots, e_{u_0v_2}, e_{u_1v_0}, \ldots, e_{u_2v_2}\}$. (6) represents the communication energy spent on the edge for sending data from a mobile to an offloading site, either and edge server or cloud server where e_{u_i,v_j} denotes the energy cost if component u is executed on site q_i and component v is executed on site q_j, and t_{u_i,v_j} is the time spent transferring data from component u on site q_i to component v on site q_j [3].

$$e_{u_i,v_j} = t_{u_i,v_j} \times p_s, \forall (u,v) \in E \text{ and } i = 0, j \in [1,2] \tag{6}$$

(7) represents the communication energy spent on the edge for receiving data from an offloading site either edge server or cloud server to a mobile device, given by

$$e_{u_i,v_j} = t_{u_i,v_j} \times p_r \forall (u,v) \in E, i \in [1,2], j = 0 \tag{7}$$

The energy a mobile device spends while waiting for data transfer between components on different offloading sites is represented by (8) taking into consideration that $p_s > p_r > p_c > p_{idle}$ [9].

$$e_{u_i,v_j} = t_{u_i,v_j} \times p_{idle}, \forall (u,v) \in E, i, j \in [1,2], i \neq j \tag{8}$$

The communication time spent to transfer data from component u on site q_i to component v on site q_j is denoted as t_{u_i,v_j} and is given by given by (9) where $d_{u,v}$ denotes the data transferred from component u to v, and $r_{i,j}$ denotes the transmission rate between sites q_i and q_j [3].

$$t_{u_i,v_j} = \frac{d_{u,v}}{r_{ij}}, \tag{9}$$

Algorithm 2 Energy-efficient multisite offloading policy algorithm

```
Input: initialization
Output: $e_M$,$e_E$,$e_C$,$t_M$,$t_E$,$t_C$

while Not at end of stages do
$R =< R_1, R_2, R_3 >$
                                        ▷ $TP_1$ for action $a = 1$ mobile
                                ▷ $TP_2$ for action $a = 2$ Edge and Cloud
    $policy = floor(2 \times rand(1, N)) + 1$
                    ▷ Random vector of 1 ( stay at mobile )and 2 ( offloading )

    $N \leftarrow 3$
    for i = 1 to N do
        for j = 1 to N do
            TP(i,j) = T(i,j,policy(i))
        end for
    end for
    $converge \leftarrow 0$
    $V_0 \leftarrow 0$
    $\gamma \leftarrow 0.9$
    while converge do
        $V = transpose(R) + \gamma \times TP \times (transpose(V_0))$
        $old\_V \leftarrow V_0$
        $V_0 \leftarrow inverse(V)$
        if abs(old_V - $V_0$) < 0.0001 then
            $converge \leftarrow 1$
        end if
    end while
    Return $< e_M, e_E, e_C, t_M, t_E, t_C >$

end while
```

4.2 Simulation results

Table 1 shows the result of the multi-site offloading simulation using MDP. The simulation was run eight times for a differing number of nodes. In the first experiment where nodes or stages is equal to 5, two of the stages have executed at the mobile device, two have executed at the edge server, and one has executed at the cloud server. The total energy consumption of the mobile device is 19.25 joules, the edge server consumes 2.19 joules, and the cloud server consumes 0.801 joules. As mentioned in a previous section, the execution starts and ends at the mobile device. Thus, there must be at least two stages executed at mobile device.

Several observations can be made when considering Tables 1–4. First, the energy consumption of executing an application on the edge server results in a larger energy saving than compared to the cloud server. Second, the time cost for multiple site execution of single edge and cloud nodes is less than the time cost for single mobile node, for example, when the nodes equal to 40, the average time cost for edge per node is 3.523 seconds, the average time cost for cloud per node is 3.839 seconds, and the average time cost for mobile per node is 10.6 seconds, as shown in Table 2. Third, the energy saving of executing an application across multiple sites (i.e., edge and cloud) is between 17.473% – 46.27% of the energy consumption of execution on single site server (i.e., mobile device), as shown in Table 3. Moreover, we noticed that the the energy saving percentage decreases as nodes increases, as shown in Table 3. However, the energy consumption of execution on a edge server is higher than a cloud server because the database is located at the cloud where there is a higher computational speed and faster access to a database, while the mobile device or edge servers have to make data requests to the cloud server. Fourth, the time cost for multiple site execution is less than the time cost for single site execution. The time cost for multiple site execution is between 41.88% – 64.52% of the time cost of execution on single site server (i.e., mobile device), as shown in Table 4.

Table 1. Energy consumption of executing an application on multiple sites.

Node	No. of execution			Energy(J)		
	Mobile	Edge	Cloud	Mobile	Edge	Cloud
5	2	2	1	19.25	2.19	0.801
10	2	6	2	18.509	6.402	2.841
15	2	10	3	19.178	12.003	3.906
20	2	12	6	18.677	12.738	7.878
25	2	14	9	19.301	15.204	10.494
30	2	16	12	19.604	19.275	15.57
35	2	21	12	19.1999	24.983	13.011
40	2	24	14	18.077	30.906	16.125

5 Conclusion

This study has investigated the problem of how to save energy and time for mobile devices by executing some components of mobile applications remotely (e.g., on the edge server or in a cloud server). A MDP-based methodology was formulated to optimize energy consumption and execution time, resulting in savings of 17.47% to nearly 46.27%.

In the future, this work will be enhanced through multiple routes:

– Various algorithms and techniques like dynamic programming or ant colony optimization (ACO) may be compared with the MDP-based model in order

Table 2. Time cost of executing an application on multiple sites.

Node	No. of execution			Time(sec.)		
	Mobile	Edge	Cloud	Mobile	Edge	Cloud
5	2	2	1	23.21	7.86	2.67
10	2	6	2	21.54	19.58	9.47
15	2	10	3	18.64	37.32	13.02
20	2	12	6	19.32	42.8	26.26
25	2	14	9	22.58	51.19	34.98
30	2	16	12	18.46	54.53	51.9
35	2	21	12	21.18	85.045	43.37
40	2	24	14	21.2	84.56	53.75

Table 3. Total energy consumption of executing an application on multiple sites and single site.

Node	Energy(J)		Energy saving (%)
	Multiple site	Single site	
5	22.241	48.065	46.27
10	27.752	93.781	29.59
15	35.087	144.126	24.34
20	39.293	190.541	20.62
25	44.999	235.441	19.11
30	54.449	284.622	19.13
35	57.1925	333.092	17.17
40	65.108	372.508	17.47

Table 4. Total time cost of executing an application on multiple sites and single site.

Node	Time (sec.)		Time saving (%)
	Multiple site	Single site	
5	33.74	52.29	64.52
10	50.59	99.94	50.62
15	68.98	149.13	46.25
20	88.38	199.68	44.26
25	108.75	251.2	43.29
30	124.79	298.39	42.83
35	149.595	353.82	42.27
40	169.51	404.71	41.88

to evaluate which algorithms perform best when optimizing computation time and energy consumption;

– The system may be expanded to be more realistic, involving multiple mobile devices, multiple edge servers, and a variety of a cloud servers leading to a more complex state space and more difficult optimization;
– Calculation of the MDP process may be varied from a centralized to a decentralized position, resulting in various impacts in optimization; and
– New reward functions, including the possible inclusion of some type of token/credit may be included [6].

References

[1] van der Meulen, R., Forni, A.A.: Gartner says demand for 4g smartphones in emerging markets spurred growth in second quarter of 2017. Technical report, Gartner (2017)

[2] Verbelen, T., Stevens, T., Simoens, P., Turck, F.D., Dhoedt, B.: Dynamic deployment and quality adaptation for mobile augmented reality applications. Journal of Systems and Software **84**(11) (2011) 1871 – 1882

[3] Terefe, M.B., Lee, H., Heo, N., Fox, G.C., Oh, S.: Energy-efficient multisite offloading policy using markov decision process for mobile cloud computing. Pervasive and Mobile Computing **27** (2016) 75–89

[4] Zhang, W., Wen, Y., Guan, K.: Energy-optimal mobile cloud computing under stochastic wireless channel. IEEE Transactions on Wireless Communications **12**(9) (September 2013) 4569 – 4581

[5] Bahl, P., Han, R.Y., Li, L.E., Satyanarayanan, M.: Advancing the state of mobile cloud computing. In: Proceedings of the Third ACM Workshop on Mobile Cloud Computing and Services. MCS '12, New York, NY, USA, ACM (2012) 21–28

[6] Nasseri, M., Alam, M., Green, R.C.: Mdp based optimal policy for collaborative processing using mobile cloud computing. IEEE 2nd International Conference on Cloud Networking (CloudNet) (2013) 123–129

[7] van Otterlo, M.: Markov decision processes: Concepts and algorithms (May 2009) Compiled for the SIKS course on Learning and Reasoning.

[8] Bellman, R.: A markovian decision process. Technical report, DTIC (1957)

[9] Kumar, K., Lu, Y.H.: Cloud computing for mobile users: Can offloading computation save energy? Computer **43**(4) (April 2010) 51–56