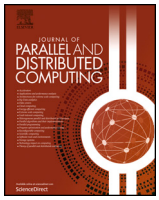
Journal of Parallel and Distributed Computing 184 (2024) 104781

Contents lists available at ScienceDirect

Journal of Parallel and Distributed Computing

journal homepage: www.elsevier.com/locate/jpdc



Scheduling independent tasks on multiple cloud-assisted edge servers with energy constraint

Keqin Li

Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

ARTICLE INFO

Keywords:
Asymptotic performance bound
Cloud-assisted edge server
Effective speed
Energy constraint
Heuristic algorithm
Mobile edge computing
Task scheduling

ABSTRACT

In this paper, we study task scheduling with or without energy constraint in mobile edge computing with multiple cloud-assisted edge servers as combinatorial optimization problems within the framework of classical scheduling theory. The first problem is to schedule a list of independent tasks on a mobile device and several heterogeneous edge servers and cloud servers, such that the makespan is minimized. The second problem is to schedule a list of independent tasks and to determine the computation and communication speeds of a mobile device, such that the makespan is minimized and the energy consumption of the mobile device does not exceed certain energy budget. The paper makes the following tangible contributions. We design heuristic task scheduling algorithms for both problems by considering the heterogeneity of computation and communication speeds. We derive a lower bound for the optimal schedule and prove an asymptotic performance bound for our heuristic algorithms. We experimentally evaluate the performance of our heuristic algorithms and show that their performance is very close to that of an optimal algorithm. Our analysis employs three key techniques, namely, the method of communication unification (i.e., all tasks have the same communication to computation ratio), the concept of effective speed of an edge server or a cloud server (i.e., the perceived speed of a server by ignoring the details and differences of communication speed and computation speed, wireless communication time and wired communication time, a regular edge server and a cloud-assisted edge server, execution time and waiting time), and the construction of virtual tasks (i.e., imaginary tasks which do not exist). Such unique techniques make it possible to derive a lower bound for the optimal solution, to derive an upper bound for the heuristic solution, to prove an asymptotic performance bound, and to find the best edge server order. To the best of the author's knowledge, this is the first paper in the literature which optimizes the makespan of task scheduling with or without energy constraint in mobile edge computing with multiple cloud-assisted edge servers.

1. Introduction

1.1. Background

Device-edge-cloud cooperative computing [9,27] provides a powerful and flexible environment for numerous applications [1,5,8,18,28,30,33] by combining various local and remote computing resources [2,4,19,22,24,31,34]. One effective way to utilize the abundant computing capabilities in such an environment is to properly schedule tasks generated on a mobile device. Task scheduling in mobile edge computing with device-edge-cloud fusion has been investigated by several researchers [6,7,10,11,13–17,23,25,26,29,32]. However, none of the existing research has studied task scheduling with device-edge-cloud collaborative computing within the traditional scheduling framework, i.e., to minimize the schedule length. A common approach adopted by many researchers is to minimize the summation of task execution times [7,10,23,26] or a weighted sum of execution time and energy consumption of all tasks [6,17,25,29].

Task scheduling for device-edge-cloud cooperative computing is a very challenging problem due to several reasons. First, mobile devices, edge servers, and cloud servers are heterogeneous in their computing speeds. Typically, an edge server has faster computing speed than a mobile device, and a cloud server has even faster computing speed. However, offloading tasks to an edge server incurs wireless communication cost, and offloading tasks to a cloud server incurs both wireless and wired communication costs. Second, there are different communication mechanisms and speeds in mobile edge computing. A mobile device communicates with an edge server via wireless communication, while an edge server communicates with a cloud server via wired communication. Typically, wired communication may have faster communication

E-mail address: lik@newpaltz.edu.

https://doi.org/10.1016/j.jpdc.2023.104781
Received 18 April 2023; Received in revised form 18 August 2023; Accepted 2 October 2023
Available online 6 October 2023

speed than wireless communication. Third, a mobile device can communicate with only one edge server at a time. This implies that there is some initial waiting time for edge servers to receive workload from a mobile device. The edge servers should be arranged in an appropriate order to reduce the impact of such waiting time on performance. Fourth, similar to the traditional scheduling theory, the performance of a heuristic scheduling algorithm should be compared with an optimal scheduling algorithm. However, it is quite challenging to obtain the knowledge of an optimal schedule due to various heterogeneities and sophisticated communications mentioned above. Yet, without comparison with an optimal algorithm, performance evaluation of a heuristic algorithm is much less interesting.

Energy consumption is another factor which makes task scheduling in mobile edge computing more difficult. A heuristic task scheduling algorithm must pay extra attention to an energy constraint, set proper computation and communication speeds for a mobile device, and make sure that the energy consumption of the mobile device does not exceed the given energy budget. Such energy constraint also makes it more troublesome to find a lower bound for an optimal solution.

Some initial attempt has been made to minimize makespan [21], where only the situation with one edge server and one cloud server was considered. The motivation of this paper is to extend the investigation to multiple edge servers and multiple cloud servers.

1.2. Contributions

In this paper, we study task scheduling with or without energy constraint in mobile edge computing with multiple cloud-assisted edge servers as combinatorial optimization problems within the framework of classical scheduling theory. The first problem is to schedule a list of independent tasks on a mobile device and several heterogeneous edge servers and cloud servers, such that the makespan is minimized. The second problem is to schedule a list of independent tasks and to determine the computation and communication speeds of a mobile device, such that the makespan is minimized and the energy consumption of the mobile device does not exceed certain energy budget. The paper makes the following tangible contributions.

- We design heuristic task scheduling algorithms for both problems by considering the heterogeneity of computation and communication speeds.
- We derive a lower bound for the optimal schedule and prove an asymptotic performance bound for our heuristic algorithms.
- We experimentally evaluate the performance of our heuristic algorithms and show that their performance is very close to that of an optimal algorithm.

Our analysis employs three key techniques (see Section 4), namely, the method of communication unification (i.e., all tasks have the same communication to computation ratio), the concept of effective speed of an edge server or a cloud server (i.e., the perceived speed of a server by ignoring the details and differences of communication speed and computation speed, wireless communication time and wired communication time, a regular edge server and a cloud-assisted edge server, execution time and waiting time), and the construction of virtual tasks (i.e., imaginary tasks which do not exist). Such unique techniques make it possible to derive a lower bound for the optimal solution, to derive an upper bound for the heuristic solution, to prove an asymptotic performance bound, and to find the best edge server order. To the best of the author's knowledge, this is the first paper in the literature which optimizes the makespan of task scheduling with or without energy constraint in mobile edge computing with multiple cloud-assisted edge servers.

We would like to emphasize that the two scheduling problems investigated in this paper are very closely related in the sense that any heuristic algorithm for task scheduling on cloud-assisted edge servers without energy constraint can be extended to task scheduling on cloud-assisted edge servers with energy constraint (see Section 6).

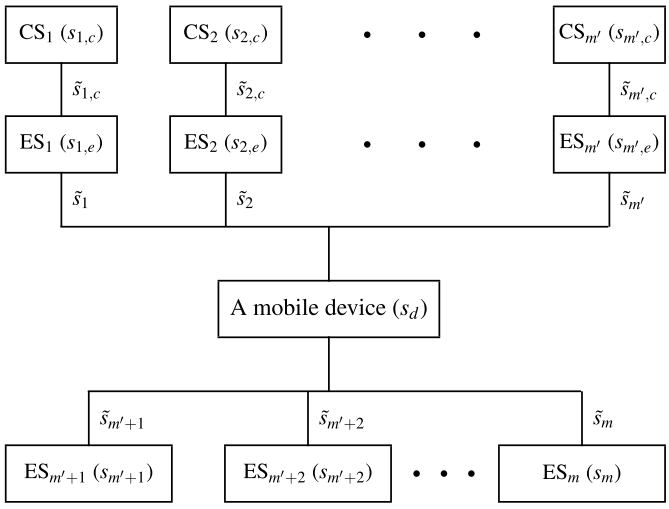


Fig. 1. A heterogeneous computing environment with device-edge-cloud fusion.

The rest of the paper is organized as follows. Sections 2–5 are concerned with task scheduling without energy constraint. In Section 2, we describe our models and problem. In Section 3, we develop our heuristic algorithms. In Section 4, we analyze the performance of our heuristic algorithm. In Section 5, we demonstrate and discuss some numerical data and experimental results. In Section 6, we study task scheduling with energy constraint. In Section 7, we give remarks on related work. In Section 8, we conclude the paper.

2. Preliminaries

In this section, we describe our models and problem.

2.1. Models

In this section, we present our computing model, task model, and execution model.

2.1.1. Computing model

We consider a heterogeneous mobile edge computing environment with a mobile device and multiple edge servers supported by cloud servers (see Fig. 1). All computation speeds are measured by billion instructions per second (Bips). All communication speeds are measured by million bits per second (Mbps).

The mobile device is the center of the computing system with device-edge-cloud fusion, which decides a task schedule and dispatches tasks to the edge servers. A mobile device is specified by its computation speed s_d.

There are m heterogeneous edge servers: ES_1, ES_2, ..., ES_m, with different communication and computation capabilities. The edge servers are classified into two categories.

The first category are *regular edge servers* (RES), for instance, $ES_{m'+1}$, $ES_{m'+2}$, ..., ES_m in Fig. 1. If ES_j is a RES, we also call it as RES_j, which is specified by $RES_j = (\tilde{s}_j, s_j)$, where $\tilde{s}_j$ is the wireless communication speed between the mobile device and ES_j, and s_j is the computation speed of ES_j.

The second category are *cloud-assisted edge servers* (CES), for instance, ES_1, ES_2, ..., $ES_{m'}$ in Fig. 1. If ES_j is a CES, we also call it as CES_j. CES_j actually includes ES_j and CS_j, where ES_j is an edge server and CS_j is a cloud server. ES_j is specified by $ES_j = (\tilde{s}_j, s_{j,e})$, where $\tilde{s}_j$ is the wireless communication speed between the mobile device and ES_j, and $s_{j,e}$ is the computation speed of ES_j. CS_j is specified

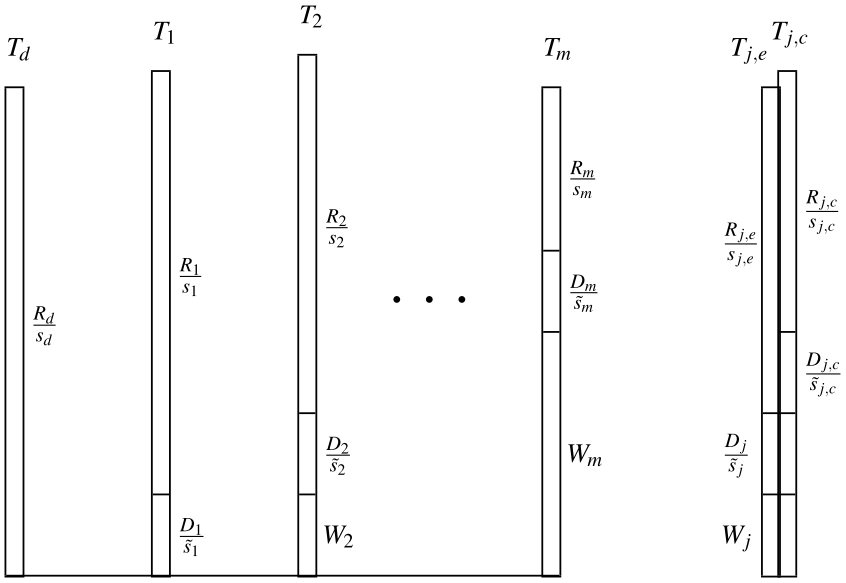


Fig. 2. A task scheduling framework on cloud-assisted edge servers.

by $\text{CS}_j = (\tilde{s}_{j,c}, s_{j,c})$, where $\tilde{s}_{j,c}$ is the wired communication speed between ES_j and CS_j, and $s_{j,c}$ is the computation speed of CS_j. CES_j is specified by $\text{CES}_j = (\tilde{s}_j, s_{j,e}, \tilde{s}_{j,c}, s_{j,c})$.

2.1.2. Task model

The mobile device has a list of independent tasks to be executed: $L = (t_1, t_2, ..., t_n)$. Task t_i is specified as $t_i = (d_i, r_i)$, where d_i is the amount of communication (MB) and r_i is the amount of computation (GI), for all $1 \le i \le n$. The ratio $\gamma_i = d_i/r_i$ gives the amount of communication per unit of computation (MB/GI). A task can be executed on the mobile device, an edge server, or a cloud server.

A schedule of L is a division of L into $(m+1)$ sublists $L_d, L_1, L_2, ..., L_m$. L_d includes tasks executed on the mobile device. Let

$$D_d = \sum_{t_i \in L_d} d_i, \text{ and } R_d = \sum_{t_i \in L_d} r_i.$$

L_j includes tasks executed on ES_j. Let

$$D_j = \sum_{t_i \in L_j} d_i, \text{ and } R_j = \sum_{t_i \in L_j} r_i.$$

If ES_j is a CES, L_j is further divided into two sublists $L_{j,e}$ and $L_{j,c}$, where $L_{j,e}$ includes tasks executed on ES_j and $L_{j,c}$ includes tasks executed on CS_j. Let

$$D_{j,e} = \sum_{t_i \in L_{j,e}} d_i,\ R_{j,e} = \sum_{t_i \in L_{j,e}} r_i,\ D_{j,c} = \sum_{t_i \in L_{j,c}} d_i, \text{ and } R_{j,c} = \sum_{t_i \in L_{j,c}} r_i.$$

It is clear that $D_j = D_{j,e} + D_{j,c}$ and $R_j = R_{j,e} + R_{j,c}$.

2.1.3. Execution model

A task scheduling framework on cloud-assisted edge servers is shown in Fig. 2. Since the mobile device can only communicate with one ES at a time, the edge servers are arranged in certain order: ES_1, ES_2, ..., ES_m, such that the mobile device sends tasks to the edge servers in this order. This means that each ES_j has certain waiting time W_j, which is the total wireless communication time of ES_1, ES_2, ..., ES_{j-1}.

If a task t_i is executed on the mobile device, its execution time is

$$\tau_i = \frac{r_i}{s_d}.$$

The total time of the mobile device is its computation time:

$$T_d = \sum_{t_i \in L_d} \tau_i = \frac{R_d}{s_d}.$$

If a task t_i is executed on a regular edge server $\text{ES}_j = \text{RES}_j$, its execution time is

$$\tau_i = \frac{d_i}{\tilde{s}_j} + \frac{r_i}{s_j}.$$

The total time of RES_j is waiting time + communication time + computation time:

$$T_j = W_j + \sum_{t_i \in L_j} \tau_i = W_j + \frac{D_j}{\tilde{s}_j} + \frac{R_j}{s_j}. \tag{1}$$

For a cloud-assisted edge server $\text{CES}_j = \text{ES}_j + \text{CS}_j$, if a task t_i is executed on ES_j, its execution time is

$$\tau_i = \frac{d_i}{\tilde{s}_j} + \frac{r_i}{s_{j,e}},$$

and if t_i is executed on CS_j, its execution time is

$$\tau_i = \frac{d_i}{\tilde{s}_j} + \frac{d_i}{\tilde{s}_{j,c}} + \frac{r_i}{s_{j,c}}.$$

The total time of ES_j is waiting time + wireless communication time + computation time:

$$T_{j,e} = W_j + \sum_{t_i \in L_j} \frac{d_i}{\tilde{s}_j} + \sum_{t_i \in L_{j,e}} \frac{r_i}{s_{j,e}} = W_j + \frac{D_j}{\tilde{s}_j} + \frac{R_{j,e}}{s_{j,e}}. \tag{2}$$

The total time of CS_j is waiting time + wireless communication time + wired communication time + computation time:

$$T_{j,c} = W_j + \sum_{t_i \in L_j} \frac{d_i}{\tilde{s}_j} + \sum_{t_i \in L_{j,c}} \frac{d_i}{\tilde{s}_{j,c}} + \sum_{t_i \in L_{j,c}} \frac{r_i}{s_{j,c}} = W_j + \frac{D_j}{\tilde{s}_j} + \frac{D_{j,c}}{\tilde{s}_{j,c}} + \frac{R_{j,c}}{s_{j,c}}. \tag{3}$$

The total time of CES_j is:

$$T_j = \max\{T_{j,e}, T_{j,c}\}. \tag{4}$$

The waiting time of ES_j is the total wireless communication time of all its predecessors:

$$W_j = \sum_{j' < j} \frac{D_{j'}}{\tilde{s}_{j'}}, \tag{5}$$

for all $1 \le j \le m$.

2.2. Problem

The *schedule length* is

$$T = \max\{T_d, T_1, T_2, ..., T_m\}. \quad (6)$$

Our combinatorial optimization problem is within the framework of traditional scheduling theory, i.e., minimizing the schedule length.

Our first combinatorial optimization problem is defined below.

Problem 1. Task Scheduling on Cloud-Assisted Edge Servers.
Input: $L = (t_1, t_2, ..., t_n)$, ES_1, ES_2, ..., ES_m.
Output: A schedule of L: $L_d, L_1, L_2, ..., L_m$, such that T is minimized.

The problem is NP-hard even for one regular edge server [21].

3. Heuristic algorithms

Algorithm 1 presents our heuristic algorithm for task scheduling on cloud-assisted edge servers.

The algorithm is essentially adapted from the classical list scheduling algorithm [12], i.e., scheduling tasks in the given order (line (9)) and assigning the next task to the earliest available device or server (lines (10), (13), (24), (27)). When a task is assigned to an edge server (lines (14), (25), (28)) whose total time is updated (lines (15), (26), (29)), the waiting times of successive edge servers are also updated (lines (16)–(18) and (21)–(23)).

Without loss of generality, we assume that the edge servers are in the order of:

$$CES_1, ..., CES_j, ..., CES_{m'}, RES_{m'+1}, ..., RES_{j'}, ..., RES_m.$$

Then, the current configuration of the computing system can be represented as:

$$\langle T_d, T_{1,e}, T_{1,c}, ..., T_{j,e}, T_{j,c}, ..., T_{m',e}, T_{m',c}, T_{m'+1}, ..., T_{j'}, ..., T_m \rangle,$$

with

$$\hat{T} = \min\{T_d, T_{1,e}, T_{1,c}, ..., T_{j,e}, T_{j,c}, ..., T_{m',e}, T_{m',c}, T_{m'+1}, ..., T_{j'}, ..., T_m\}.$$

If $T_d = \hat{T}$, then the new configuration is (line (12))

$$\langle T_d + r_i/s_d, T_{1,e}, T_{1,c}, ..., T_{j,e}, T_{j,c}, ..., T_{m',e}, T_{m',c}, T_{m'+1}, ..., T_{j'}, ..., T_m \rangle.$$

If $T_{j,e} = \hat{T}$, then the new configuration is (lines (20)–(23), (26))

$$\langle T_d, T_{1,e}, T_{1,c}, ..., T_{j,e} + d_i/\tilde{s}_j + r_i/s_{j,e}, T_{j,c} + d_i/\tilde{s}_j, ..., T_{m',e} + d_i/\tilde{s}_j, T_{m',c} + d_i/\tilde{s}_j, T_{m'+1} + d_i/\tilde{s}_j, ..., T_{j'} + d_i/\tilde{s}_j, ..., T_m + d_i/\tilde{s}_j \rangle.$$

If $T_{j,c} = \hat{T}$, then the new configuration is (lines (20)–(23), (29))

$$\langle T_d, T_{1,e}, T_{1,c}, ..., T_{j,e} + d_i/\tilde{s}_j, T_{j,c} + d_i/\tilde{s}_j + d_i/\tilde{s}_{j,c} + r_i/s_{j,c}, ..., T_{m',e} + d_i/\tilde{s}_j, T_{m',c} + d_i/\tilde{s}_j, T_{m'+1} + d_i/\tilde{s}_j, ..., T_{j'} + d_i/\tilde{s}_j, ..., T_m + d_i/\tilde{s}_j \rangle.$$

If $T_{j'} = \hat{T}$, then the new configuration is (lines (15)–(18))

$$\langle T_d, T_{1,e}, T_{1,c}, ..., T_{j,e}, T_{j,c}, ..., T_{m',e}, T_{m',c}, T_{m'+1}, ..., T_{j'} + d_i/\tilde{s}_{j'} + r_i/s_{j'}, ..., T_m + d_i/\tilde{s}_{j'} \rangle.$$

In addition to Algorithm 1, we also consider Algorithm 2, which follows the same framework of Algorithm 1, except that when task t_i is being scheduled, we calculate the total time of the mobile device and each server:

$$T'_d = T_d + r_i/s_d,$$
$$T'_j = T_j + d_i/\tilde{s}_j + r_i/s_j,$$
$$T'_{j,e} = T_{j,e} + d_i/\tilde{s}_j + r_i/s_{j,e},$$
$$T'_{j,c} = T_{j,c} + d_i/\tilde{s}_j + d_i/\tilde{s}_{j,c} + r_i/s_{j,c},$$

Algorithm 1 Heuristic Algorithm for Task Scheduling on Cloud-Assisted Edge Servers.

Input: $L = (t_1, t_2, ..., t_n)$, ES_1, ES_2, ..., ES_m.
Output: A schedule of L: $L_d, L_1, L_2, ..., L_m$, such that T is minimized.

```
(1)  L_d ← ∅, T_d ← 0;
(2)  for (j = 1; j ≤ m; j++) do
(3)      if (ES_j is a RES) then
(4)          L_j ← ∅, T_j ← 0;
(5)      else //ES_j is a CES
(6)          L_{j,e} ← ∅, L_{j,c} ← ∅, T_{j,e} ← 0, T_{j,c} ← 0;
(7)      end if
(8)  end do;
(9)  for (i = 1; i ≤ n; i++) do
(10)     if (T_d = min{T_d, T_1, T_2, ..., T_m}) then
(11)         add t_i to L_d;
(12)         T_d ← T_d + r_i/s_d;
(13)     else if (T_j = min{T_d, T_1, T_2, ..., T_m} and ES_j is a RES) then
(14)         add t_i to L_j;
(15)         T_j ← T_j + d_i/s̃_j + r_i/s_j;
(16)         for (j' = j + 1; j' ≤ m; j'++) do
(17)             T_{j'} ← T_{j'} + d_i/s̃_j;
(18)         end do
(19)     else //ES_j is a CES
(20)         T_{j,e} ← T_{j,e} + d_i/s̃_j, T_{j,c} ← T_{j,c} + d_i/s̃_j;
(21)         for (j' = j + 1; j' ≤ m; j'++) do
(22)             T_{j'} ← T_{j'} + d_i/s̃_j;
(23)         end do;
(24)         if (T_{j,e} = min{T_d, T_1, T_2, ..., T_m}) then
(25)             add t_i to L_{j,e};
(26)             T_{j,e} ← T_{j,e} + r_i/s_{j,e};
(27)         else //T_{j,c} = min{T_d, T_1, T_2, ..., T_m}
(28)             add t_i to L_{j,c};
(29)             T_{j,c} ← T_{j,c} + d_i/s̃_{j,c} + r_i/s_{j,c};
(30)         end if
(31)     end if
(32) end do.
```

and add t_i to the mobile device or the edge server or the cloud server which has the minimum total time if t_i is added.

The time complexity of both Algorithms 1 and 2 is $O(mn)$, since $O(m)$ time is spent for each task.

It should be emphasized that although not mentioned in Algorithm 1, it is an important component of a heuristic algorithm to decide the order of the edge servers, which affects the schedule length. Some reasonable and promising edge server orders are as follows.

- $\bar{s}_1 \geq \bar{s}_2 \geq \cdots \geq \bar{s}_m$: $\bar{s}_j$ is the effective computation speed of ES_j to be introduced in Section 4.1. An edge server with larger $\bar{s}_j$ should start its execution earlier to complete more work.
- $\phi_1 \leq \phi_2 \leq \cdots \geq \phi_m$: ϕ_j is the percentage of the communication time in the execution time of ES_j to be introduced in Section 4.1. An edge server with smaller ϕ_j should start its execution earlier to reduce the waiting time of subsequent edge servers.
- $\bar{s}_1/\phi_1 \geq \bar{s}_2/\phi_2 \geq \cdots \geq \bar{s}_m/\phi_m$: This is a combination of the above two orders and will be adopted by our algorithm (see Section 4.2 for justification).
- $\tilde{s}_1 \geq \tilde{s}_2 \geq \cdots \geq \tilde{s}_m$: An edge server with larger $\tilde{s}_j$ should start its execution earlier to reduce the waiting time of subsequent edge servers. It turns out that this is actually equivalent to $\bar{s}_1/\phi_1 \geq \bar{s}_2/\phi_2 \geq \cdots \geq \bar{s}_m/\phi_m$ (see Section 4.2 for explanation).

4. Analysis

In this section, we analyze the performance of our heuristic algorithm. We derive a lower bound for the optimal solution and an upper bound for the heuristic solution, and then prove a performance bound.

4.1. Effective speed

To derive a lower bound for the optimal schedule length (Theorem 1) and an upper bound for the heuristic schedule length (Theo-

rem 2), we consider a special case where all tasks have the same communication to computation ratio, $\gamma_1 = \gamma_2 = \cdots = \gamma_n = \gamma$, namely, $d_i = \gamma r_i$, for all $1 \le i \le n$. Such *communication unification* helps to analyze the performance of our heuristic algorithm (Theorem 3), where the assumption of unified γ is removed.

For ease of analysis, we introduce the concept of *effective speed* of a server as perceived from an outside observer. Assume that for a CES_j, $T_j = T_{j,e} = T_{j,c}$, that is, for some C_j, we have

$$\frac{R_{j,e}}{s_{j,e}} = \frac{D_{j,c}}{\tilde{s}_{j,c}} + \frac{R_{j,c}}{s_{j,c}} = C_j.$$

From

$$\frac{R_{j,e}}{s_{j,e}} = C_j,$$

we get

$$R_{j,e} = s_{j,e} C_j.$$

Since

$$\frac{D_{j,c}}{\tilde{s}_{j,c}} + \frac{R_{j,c}}{s_{j,c}} = \left(\frac{\gamma}{\tilde{s}_{j,c}} + \frac{1}{s_{j,c}}\right) R_{j,c} = \left(\frac{\gamma s_{j,c} + \tilde{s}_{j,c}}{\tilde{s}_{j,c} s_{j,c}}\right) R_{j,c} = \frac{R_{j,c}}{\bar{s}_{j,c}} = C_j,$$

we get

$$R_{j,c} = \bar{s}_{j,c} C_j,$$

where

$$\bar{s}_{j,c} = \frac{\tilde{s}_{j,c} s_{j,c}}{\gamma s_{j,c} + \tilde{s}_{j,c}} \tag{7}$$

is the *effective computation speed* of CS_j, i.e., the perceived computation speed of CS_j by ignoring the details and differences of communication speed $\tilde{s}_{j,c}$ and computation speed $s_{j,c}$. Using $\bar{s}_{j,c}$, $T_{j,c}$ can be rewritten (actually simplified) as

$$T_{j,c} = W_j + \frac{D_j}{\tilde{s}_j} + \frac{R_{j,c}}{\bar{s}_{j,c}}.$$

Furthermore, we have

$$R_j = R_{j,e} + R_{j,c} = (s_{j,e} + \bar{s}_{j,c}) C_j = s_j C_j,$$

where

$$s_j = s_{j,e} + \bar{s}_{j,c} = s_{j,e} + \frac{\tilde{s}_{j,c} s_{j,c}}{\gamma s_{j,c} + \tilde{s}_{j,c}} = \frac{\gamma s_{j,e} s_{j,c} + s_{j,e} \tilde{s}_{j,c} + \tilde{s}_{j,c} s_{j,c}}{\gamma s_{j,c} + \tilde{s}_{j,c}} \tag{8}$$

is the *effective computation speed* of CES_j, i.e., the perceived computation speed of CES_j by ignoring the details and differences of ES_j and CS_j. Using s_j, T_j can be rewritten (actually simplified) as

$$T_j = W_j + \frac{D_j}{\tilde{s}_j} + \frac{R_j}{s_j}. \tag{9}$$

Now, for all ES_j, no matter whether it is a RES or a CES, we have

$$T_j = W_j + \frac{D_j}{\tilde{s}_j} + \frac{R_j}{s_j} = W_j + \left(\frac{\gamma}{\tilde{s}_j} + \frac{1}{s_j}\right) R_j = W_j + \left(\frac{\gamma s_j + \tilde{s}_j}{\tilde{s}_j s_j}\right) R_j = W_j + \frac{R_j}{\bar{s}_j}, \tag{10}$$

where

$$\bar{s}_j = \frac{\tilde{s}_j s_j}{\gamma s_j + \tilde{s}_j} \tag{11}$$

for a RES, and

$$\bar{s}_j = \frac{\tilde{s}_j s_j}{\gamma s_j + \tilde{s}_j} = \frac{\tilde{s}_j (\gamma s_{j,e} s_{j,c} + s_{j,e} \tilde{s}_{j,c} + \tilde{s}_{j,c} s_{j,c})}{\gamma^2 s_{j,e} s_{j,c} + \gamma s_{j,e} \tilde{s}_{j,c} + \gamma \tilde{s}_{j,c} s_{j,c} + \gamma \tilde{s}_j s_{j,c} + \tilde{s}_j \tilde{s}_{j,c}} \tag{12}$$

for a CES, is the *effective execution speed* of ES_j, i.e., the perceived computation speed of ES_j, by ignoring the difference between RES or CES and ignoring the details of wireless communication time, wired communication time, and computation time. We call the quantity

$$T'_j = \frac{D_j}{\tilde{s}_j} + \frac{R_j}{s_j} = \frac{R_j}{\bar{s}_j} \tag{13}$$

as the *execution time* (including all wireless communication time, wired communication time, and computation time) of ES_j, whether in the presence or absence of a cloud server.

Let us define

$$\phi_j = \gamma \frac{\bar{s}_j}{\tilde{s}_j}, \tag{14}$$

which is

$$\phi_j = \gamma \frac{\bar{s}_j}{\tilde{s}_j} = \frac{\gamma s_j}{\gamma s_j + \tilde{s}_j} \tag{15}$$

for a RES, and

$$\phi_j = \gamma \frac{\bar{s}_j}{\tilde{s}_j} = \frac{\gamma s_j}{\gamma s_j + \tilde{s}_j} = \frac{\gamma^2 s_{j,e} s_{j,c} + \gamma s_{j,e} \tilde{s}_{j,c} + \gamma \tilde{s}_{j,c} s_{j,c}}{\gamma^2 s_{j,e} s_{j,c} + \gamma s_{j,e} \tilde{s}_{j,c} + \gamma \tilde{s}_{j,c} s_{j,c} + \gamma \tilde{s}_j s_{j,c} + \tilde{s}_j \tilde{s}_{j,c}} \tag{16}$$

for a CES, for all $1 \le j \le m$. It is easy to see that ϕ_j is actually

$$\phi_j = \frac{\frac{D_j}{\tilde{s}_j}}{\frac{D_j}{\tilde{s}_j} + \frac{R_j}{s_j}} = \frac{\frac{D_j}{\tilde{s}_j}}{\frac{R_j}{\bar{s}_j}}, \tag{17}$$

i.e., the percentage of the wireless communication time in the execution time of ES_j.

4.2. Optimal solution

To derive a lower bound for the optimal solution, let us consider an ideal case:

$$T_d = T_1 = T_2 = \cdots = T_m = T, \tag{18}$$

with

$$T_j = T_{j,e} = T_{j,c},$$

where

$$T_d = \frac{R_d}{s_d},$$

$$T_j = W_j + T'_j = \sum_{j' < j} \frac{D_{j'}}{\tilde{s}_{j'}} + \frac{R_j}{\bar{s}_j}, \quad 1 \le j \le m.$$

It is clear that even an optimal schedule may not achieve such an ideal situation, where all mobile devices, edge servers, and cloud servers complete their tasks at the same time. Thus, T can serve as a lower bound for the optimal solution.

To pave the way for further discussion, we start with some calculations. Since

$$T_d = \frac{R_d}{s_d} = T_1 = \frac{R_1}{\bar{s}_1},$$

we get

$$R_1 = \left(\frac{\bar{s}_1}{s_d}\right) R_d.$$

Since

$$T_d = \frac{R_d}{s_d} = T_2 = \frac{D_1}{\tilde{s}_1} + \frac{R_2}{\bar{s}_2},$$

we get

$$R_2 = \bar{s}_2\left(\frac{R_d}{s_d} - \frac{D_1}{\bar{s}_1}\right)$$
$$= \bar{s}_2\left(\frac{R_d}{s_d} - \frac{\gamma R_1}{\bar{s}_1}\right)$$
$$= \bar{s}_2\left(\frac{R_d}{s_d} - \frac{\gamma}{\bar{s}_1}\left(\frac{\bar{s}_1}{s_d}\right)R_d\right)$$
$$= \frac{\bar{s}_2}{s_d}(1-\phi_1)R_d.$$

Since

$$T_d = \frac{R_d}{s_d} = T_3 = \frac{D_1}{\bar{s}_1} + \frac{D_2}{\bar{s}_2} + \frac{R_3}{\bar{s}_3},$$

we get

$$R_3 = \bar{s}_3\left(\frac{R_d}{s_d} - \frac{D_1}{\bar{s}_1} - \frac{D_2}{\bar{s}_2}\right)$$
$$= \bar{s}_3\left(\frac{R_d}{s_d} - \frac{\gamma R_1}{\bar{s}_1} - \frac{\gamma R_2}{\bar{s}_2}\right)$$
$$= \bar{s}_3\left(\frac{R_d}{s_d} - \frac{\gamma}{\bar{s}_1}\left(\frac{\bar{s}_1}{s_d}\right)R_d - \frac{\gamma}{\bar{s}_2}\cdot\frac{\bar{s}_2}{s_d}(1-\phi_1)R_d\right)$$
$$= \frac{\bar{s}_3}{s_d}(1-\phi_1-\phi_2(1-\phi_1))R_d$$
$$= \frac{\bar{s}_3}{s_d}(1-\phi_1)(1-\phi_2)R_d.$$

Since

$$T_d = \frac{R_d}{s_d} = T_4 = \frac{D_1}{\bar{s}_1} + \frac{D_2}{\bar{s}_2} + \frac{D_3}{\bar{s}_3} + \frac{R_4}{\bar{s}_4},$$

we get

$$R_4 = \bar{s}_4\left(\frac{R_d}{s_d} - \frac{D_1}{\bar{s}_1} - \frac{D_2}{\bar{s}_2} - \frac{D_3}{\bar{s}_3}\right)$$
$$= \bar{s}_4\left(\frac{R_d}{s_d} - \frac{\gamma R_1}{\bar{s}_1} - \frac{\gamma R_2}{\bar{s}_2} - \frac{\gamma R_3}{\bar{s}_3}\right)$$
$$= \bar{s}_4\left(\frac{R_d}{s_d} - \frac{\gamma}{\bar{s}_1}\left(\frac{\bar{s}_1}{s_d}\right)R_d - \frac{\gamma}{\bar{s}_2}\cdot\frac{\bar{s}_2}{s_d}(1-\phi_1)R_d - \frac{\gamma}{\bar{s}_3}\cdot\frac{\bar{s}_3}{s_d}(1-\phi_1)(1-\phi_2)R_d\right)$$
$$= \frac{\bar{s}_4}{s_d}(1-\phi_1-\phi_2(1-\phi_1)-\phi_3(1-\phi_1)(1-\phi_2))R_d$$
$$= \frac{\bar{s}_4}{s_d}(1-\phi_1)(1-\phi_2)(1-\phi_3)R_d.$$

The above calculations inspire us to define

$$Q_j = \prod_{j'<j}(1-\phi_{j'}) = \prod_{j'<j}\frac{\tilde{s}_j}{\gamma s_j + \tilde{s}_j}. \tag{19}$$

Q_j is actually the percentage of the execution time

$$T'_j = (1-\phi_{j-1})T'_{j-1} = \cdots = (1-\phi_{j-1})\cdots(1-\phi_1)T'_1 = Q_j T \tag{20}$$

in the total time $T_j = T$ of ES_j. Then, it can be verified (e.g., by an inductive proof) that

$$\frac{R_j}{\bar{s}_j} = Q_j\frac{R_d}{s_d},$$

or

$$R_j = \frac{\bar{s}_j}{s_d}Q_j R_d,$$

or

$$\frac{R_j}{Q_j\bar{s}_j} = \frac{R_d}{s_d},$$

for all $1 \le j \le m$. $Q_j\bar{s}_j$ can be regarded as the *effective execution speed of ES_j with waiting time included*. Due to the waiting time W_j, the perceived execution speed of ES_j is reduced by a factor of Q_j.

The above discussion leads to the following theorem, which gives a lower bound for the optimal schedule length $T^*(L)$ of L.

Theorem 1. *If $d_i = \gamma r_i$, for all $1 \le i \le n$, the optimal schedule length $T^*(L)$ has the following lower bound:*

$$T^*(L) \ge \frac{R}{s_d + \sum_{j=1}^{m} Q_j\bar{s}_j}. \tag{21}$$

Proof. It is clear that $T^*(L) \ge T$, where T is the identical completion time of all mobile devices, edge servers, and cloud servers. To find T, we notice that

$$R_d + \sum_{j=1}^{m} R_j = R,$$

that is,

$$R_d + \sum_{j=1}^{m}\frac{\bar{s}_j}{s_d}Q_j R_d = R,$$

from which we obtain

$$R_d = \frac{R}{1 + \sum_{j=1}^{m}\frac{\bar{s}_j}{s_d}Q_j},$$

and

$$R_j = \frac{\frac{\bar{s}_j}{s_d}Q_j}{1 + \sum_{j=1}^{m}\frac{\bar{s}_j}{s_d}Q_j}R,$$

and

$$T = \frac{R_d}{s_d} = \frac{R_j}{Q_j\bar{s}_j} = \frac{R}{s_d + \sum_{j=1}^{m}Q_j\bar{s}_j} = \frac{R}{S},$$

where

$$S = s_d + \sum_{j=1}^{m}Q_j\bar{s}_j \tag{22}$$

is the *aggregated execution speed* of the device-edge-cloud collaborative computing system. The proof of the theorem is completed. □

We would like to mention that the above theorem includes the lower bound in [21] for one CES as a special case.

Notice that T in the proof Theorem 1 depends on the order of the edge servers in our task scheduling framework. We would like to mention that T is minimized when the edge servers are arranged in such an order that

$$\tilde{s}_1 \ge \tilde{s}_2 \ge \cdots \ge \tilde{s}_m. \tag{23}$$

To show this, we need to find an edge server order which maximizes the aggregated execution speed and minimizes T. Assume that the edge servers are in an order:

$$\mathrm{ES}_1, \ldots, \mathrm{ES}_{j'-1}, \mathrm{ES}_{j'}, \mathrm{ES}_{j'+1}, \mathrm{ES}_{j'+2}, \ldots, \mathrm{ES}_m,$$

with aggregated execution speed:

$$S = \sum_{j=1}^{j'-1}Q_j\bar{s}_j + Q_{j'}\bar{s}_{j'} + Q_{j'}(1-\phi_{j'})\bar{s}_{j'+1} + \sum_{j=j'+2}^{m}Q_j\bar{s}_j.$$

We exchange the order of $\mathrm{ES}_{j'}$ and $\mathrm{ES}_{j'+1}$ and get another order:

$$\mathrm{ES}_1, \ldots, \mathrm{ES}_{j'-1}, \mathrm{ES}_{j'+1}, \mathrm{ES}_{j'}, \mathrm{ES}_{j'+2}, \ldots, \mathrm{ES}_m,$$

with aggregated execution speed:

$$S' = \sum_{j=1}^{j'-1} Q_j \bar{s}_j + Q_{j'} \bar{s}_{j'+1} + Q_{j'}(1-\phi_{j'+1})\bar{s}_{j'} + \sum_{j=j'+2}^{m} Q_j \bar{s}_j.$$

Notice that

$$S' - S = Q_{j'}(\phi_{j'} \bar{s}_{j'+1} - \phi_{j'+1}\bar{s}_{j'}),$$

which means that $S' \geq S$ if and only if $\phi_{j'} \bar{s}_{j'+1} - \phi_{j'+1}\bar{s}_{j'} \geq 0$, that is, $\phi_{j'} \bar{s}_{j'+1} \geq \phi_{j'+1}\bar{s}_{j'}$, or,

$$\frac{\bar{s}_{j'+1}}{\phi_{j'+1}} \geq \frac{\bar{s}_{j'}}{\phi_{j'}}.$$

Since

$$\frac{\bar{s}_j}{\phi_j} = \frac{\tilde{s}_j}{\gamma},$$

the last inequality is equivalent to $\tilde{s}_{j'+1} \geq \tilde{s}_{j'}$. Since T is minimized if and only if the aggregated execution speed is maximized, we know that T is minimized if and only if $\tilde{s}_1 \geq \tilde{s}_2 \geq \cdots \geq \tilde{s}_m$, that is, the edge servers are arranged in the decreasing order of wireless communication speeds.

4.3. Heuristic solution

The main result of this section is the following theorem, which gives an upper bound for the heuristic schedule length $T(L)$ of L.

Theorem 2. *If $d_i = \gamma r_i$, for all $1 \leq i \leq n$, the heuristic schedule length $T(L)$ has the following upper bound:*

$$T(L) \leq \frac{R}{s_d + \sum_{j=1}^{m} Q_j \bar{s}_j} + \frac{s_d + \sum_{j=1}^{m} \bar{s}_j}{s_d + \sum_{j=1}^{m} Q_j \bar{s}_j}\tau^*, \tag{24}$$

where τ^* is the longest execution time of any task anywhere.

Proof. Let the heuristic schedule length be

$$T(L) = \max\{T_d, T_1, T_2, ..., T_m\}. \tag{25}$$

Assume that t_i is the last task completed at time $T(L)$. It is clear that all mobile devices, edge servers, and cloud servers are busy (not available) up to time $T(L) - \tau_i$, i.e., $T_d \geq T(L) - \tau_i$, and $T_j \geq T(L) - \tau_i$, for all $1 \leq j \leq m$.

For the mobile device, we have

$$R_d = s_d T_d \geq s_d(T(L) - \tau_i).$$

For edge servers, we notice that

$$R_j \geq \bar{s}_j(T(L) - \tau_i - W_j),$$

for all $1 \leq j \leq m$, no matter whether ES_j is a RES or a CES. If ES_j is a RES, we have

$$R_j = \bar{s}_j T'_j = \bar{s}_j(T_j - W_j) \geq \bar{s}_j(T(L) - \tau_i - W_j).$$

If ES_j is a CES and $T_{j,e} = T_{j,c}$, the above argument is also valid. The complication comes from the situation where $T_{j,e} \neq T_{j,c}$. In this case, we can imagine that the tasks are divisible and transferable from ES_j to CS_j if $T_{j,e} > T_{j,c}$, or from CS_j to ES_j if $T_{j,e} < T_{j,c}$, such that the new total times are

$$\tilde{T}_{j,e} = \tilde{T}_{j,c} = \tilde{T}_j \geq \min\{T_{j,e}, T_{j,c}\} \geq T(L) - \tau_i.$$

It is clear that

$$R_j = \bar{s}_j(\tilde{T}_j - W_j) \geq \bar{s}_j(T(L) - \tau_i - W_j).$$

We can show by induction that

$$W_j \leq (1 - Q_j)T(L),$$

for all $1 \leq j \leq m$. The base cases for $j = 1, 2$ are trivial: $W_1 = 0 = (1 - Q_1)T(L)$ and $W_2 = \phi_1 T_1 \leq \phi_1 T(L) = (1 - Q_2)T(L)$. For the general case, we have

$$\begin{aligned}
W_j &= W_{j-1} + \phi_{j-1}T'_{j-1} \quad \text{(by the definition of waiting time)} \\
&= W_{j-1} + \phi_{j-1}(T_{j-1} - W_{j-1}) \quad \text{(by the definition of execution time)} \\
&= (1 - \phi_{j-1})W_{j-1} + \phi_{j-1}T_{j-1} \quad \text{(straightforward)} \\
&\leq (1 - \phi_{j-1})(1 - Q_{j-1})T(L) \\
&\quad + \phi_{j-1}T_{j-1} \quad \text{(by the induction hypothesis)} \\
&\leq (1 - \phi_{j-1})(1 - Q_{j-1})T(L) + \phi_{j-1}T(L) \quad \text{(by the definition of } T(L)) \\
&= (1 - Q_j)T(L). \quad \text{(by the definition of } Q_j)
\end{aligned}$$

Therefore, we get

$$R_j \geq \bar{s}_j(T(L) - \tau_i - (1 - Q_j)T(L)) = \bar{s}_j(Q_j T(L) - \tau_i),$$

for all $1 \leq j \leq m$. Finally, since

$$R = R_d + \sum_{j=1}^{m} R_j \geq s_d(T(L) - \tau_i) + \sum_{j=1}^{m} \bar{s}_j(Q_j T(L) - \tau_i),$$

that is,

$$R \geq \left(s_d + \sum_{j=1}^{m} Q_j \bar{s}_j\right)T(L) - \left(s_d + \sum_{j=1}^{m} \bar{s}_j\right)\tau_i,$$

which gives

$$T(L) \leq \frac{R}{s_d + \sum_{j=1}^{m} Q_j \bar{s}_j} + \frac{s_d + \sum_{j=1}^{m} \bar{s}_j}{s_d + \sum_{j=1}^{m} Q_j \bar{s}_j}\tau^*.$$

The theorem is proved. □

We would like to mention that the above theorem includes the upper bound in [21] for one CES as a special case.

4.4. Performance bound

We say that a heuristic algorithm has an *asymptotic performance bound B* if

$$\lim_{T^*(L)/\tau^* \to \infty} \frac{T(L)}{T^*(L)} = B. \tag{26}$$

Consider two task lists: $L = (t_1, t_2, ..., t_n)$, where $t_i = (d_i, r_i)$, and $L' = (t'_1, t'_2, ..., t'_n)$, where $t'_i = (d'_i, r'_i)$. If $d_i \leq d'_i$ and $r_i \leq r'_i$, then any schedule of L' is also applicable to L, simply executing t_i in the same time slot of t'_i on the same device or server, with some (wireless and wired) communication time and computation time possibly unused. However, it is very likely that L can have a better schedule with shorter schedule length. This implies that for optimal schedules, we have $T^*(L) \leq T^*(L')$, and for heuristic schedules, we have $T(L) \leq T(L')$.

For a list of tasks $L = (t_1, t_2, ..., t_n)$, where $t_i = (d_i, r_i)$ and $\gamma_i = d_i/r_i$, we define

$$\gamma' = \min\{\gamma_1, \gamma_2, ..., \gamma_n\}, \tag{27}$$

and

$$\gamma'' = \max\{\gamma_1, \gamma_2, ..., \gamma_n\}. \tag{28}$$

Let the aggregated execution speed $S(\gamma)$ of the device-edge-cloud collaborative computing system be a function of γ.

The following theorem gives an asymptotic performance bound for our heuristic algorithm for task scheduling on cloud-assisted edge servers.

Theorem 3. *Our heuristic algorithm for task scheduling on cloud-assisted edge servers has an asymptotic performance bound*

$$\frac{S(\gamma')}{S(\gamma'')}. \tag{29}$$

Proof. Let us construct $L' = (t'_1, t'_2, ..., t'_n)$ with $t'_i = (\gamma' r_i, r_i)$, and $L'' = (t''_1, t''_2, ..., t''_n)$ with $t''_i = (\gamma'' r_i, r_i)$, where t'_i and t''_i are *virtual tasks* introduced for the purpose of analysis. It is clear that

$$T^*(L') \le T^*(L) \le T(L) \le T(L'').$$

From Theorem 1, we know that

$$T^*(L') \ge \frac{R}{S(\gamma')}.$$

From Theorem 2, we know that

$$T(L'') \le \frac{R}{S(\gamma'')} + \frac{1}{S(\gamma'')}\left(s_d + \sum_{j=1}^{m} \bar{s}_j\right)\tau^*.$$

Combining all the above inequalities, we reach

$$T(L) \le \frac{S(\gamma')}{S(\gamma'')} T^*(L) + \frac{1}{S(\gamma'')}\left(s_d + \sum_{j=1}^{m} \bar{s}_j\right)\tau^*,$$

and

$$\lim_{T^*(L)/\tau^* \to \infty} \frac{T(L)}{T^*(L)} = \frac{S(\gamma')}{S(\gamma'')}.$$

This proves the theorem. □

We would like to mention that the above theorem includes the asymptotic performance bound in [21] for one CES as a special case.

5. Performance evaluation

5.1. Parameter setting

We consider a heterogeneous computing environment with seven computing units: one mobile device with $s_d = 1.5$ Bips, two cloud-assisted edge servers CES_1 (ES_1 + CS_1) and CES_2 (ES_2 + CS_2), and two regular edge servers RES_3 and RES_4, with the following parameter setting [3]:

CES_1 : $\tilde{s}_1 = 50$ Mbps, $s_{1,e} = 2.5$ Bips, $\tilde{s}_{1,c} = 95$ Mbps, $s_{1,c} = 3.5$ Bips,

CES_2 : $\tilde{s}_2 = 45$ Mbps, $s_{2,e} = 2.6$ Bips, $\tilde{s}_{2,c} = 90$ Mbps, $s_{2,c} = 3.6$ Bips,

RES_3 : $\tilde{s}_3 = 40$ Mbps, $s_{3,e} = 2.7$ Bips,

RES_4 : $\tilde{s}_3 = 35$ Mbps, $s_{3,e} = 2.8$ Bips.

The edge servers have been arranged in the order: $\tilde{s}_1 > \tilde{s}_2 > \tilde{s}_3 > \tilde{s}_4$.

L is a list of random tasks. The r_i's are independent and identically distributed random variables uniformly distributed in the interval $[1.0, 4.0]$ GI. The γ_i's are independent and identically distributed random variables uniformly distributed in the interval $[\gamma', \gamma'']$, where $\gamma' = 1.0$ MB/GI and $\gamma'' = 5.0$ MB/GI.

5.2. Numerical data

In this section, we demonstrate and discuss some numerical data.

In Table 1, we show the effective computation speed s_j, the effective execution speed $\bar{s}_j$, the percentage of the wireless communication time in the execution time ϕ_j, the percentage of the execution time in the total time Q_j, and the effective execution speed with waiting time included $Q_j \bar{s}_j$, for all edge servers, as well as the aggregated execution speed S, for $\gamma = 0, 1, 2, 3, 4, 5$. It is easily observed that as γ increases, s_j, $\bar{s}_j$, Q_j, $Q_j \bar{s}_j$, and S decrease significantly, and ϕ_j increases noticeably.

Table 1
Effective speeds of edge servers.

j	s_j	$\bar{s}_j$	ϕ_j	Q_j	$Q_j \bar{s}_j$
$\gamma = 0.00000, S = 19.20000$					
1	6.00000	6.00000	0.00000	1.00000	6.00000
2	6.20000	6.20000	0.00000	1.00000	6.20000
3	2.70000	2.70000	0.00000	1.00000	2.70000
4	2.80000	2.80000	0.00000	1.00000	2.80000
$\gamma = 1.00000, S = 15.44792$					
1	5.87563	5.25778	0.10516	1.00000	5.25778
2	6.06154	5.34197	0.11871	0.89484	4.78023
3	2.70000	2.52927	0.06323	0.78862	1.99463
4	2.80000	2.59259	0.07407	0.73875	1.91528
$\gamma = 2.00000, S = 12.89502$					
1	5.75980	4.68127	0.18725	1.00000	4.68127
2	5.93333	4.69519	0.20868	0.81275	3.81601
3	2.70000	2.37885	0.11894	0.64315	1.52996
4	2.80000	2.41379	0.13793	0.56665	1.36778
$\gamma = 3.00000, S = 11.06848$					
1	5.65166	4.22049	0.25323	1.00000	4.22049
2	5.81429	4.19012	0.27934	0.74677	3.12906
3	2.70000	2.24532	0.16840	0.53817	1.20836
4	2.80000	2.25806	0.19355	0.44754	1.01057
$\gamma = 4.00000, S = 9.70912$					
1	5.55046	3.84371	0.30750	1.00000	3.84371
2	5.70345	3.78470	0.33642	0.69250	2.62092
3	2.70000	2.12598	0.21260	0.45953	0.97696
4	2.80000	2.12121	0.24242	0.36184	0.76753
$\gamma = 5.00000, S = 8.66492$					
1	5.45556	3.52983	0.35298	1.00000	3.52983
2	5.60000	3.45205	0.38356	0.64702	2.23354
3	2.70000	2.01869	0.25234	0.39885	0.80515
4	2.80000	2.00000	0.28571	0.29820	0.59640

Table 2
Experimental results for task scheduling on cloud-assisted edge servers.

	$\tilde{s}_1 < \tilde{s}_2 < \tilde{s}_3 < \tilde{s}_4$		$\tilde{s}_1 > \tilde{s}_2 > \tilde{s}_3 > \tilde{s}_4$	
n	Algorithm 1	Algorithm 2	Algorithm 1	Algorithm 2
20	1.66089	1.61996	1.61772	1.58679
40	1.55105	1.52683	1.50824	1.49046
60	1.51877	1.49742	1.46923	1.45343
80	1.50358	1.48324	1.44964	1.43604
100	1.49121	1.47375	1.44108	1.42828
120	1.48343	1.46744	1.43348	1.42047
140	1.47974	1.46355	1.42658	1.41601
160	1.47395	1.45823	1.42394	1.41240
180	1.47151	1.45661	1.42001	1.40921
200	1.46946	1.45481	1.41803	1.40732

In Fig. 3, we display the aggregated execution speed $S(\gamma)$ as a function of γ. It is clear that $S(\gamma)$ is a decreasing function of γ.

In Fig. 4, we display the asymptotic performance bound $S(\gamma')/S(\gamma'')$ as a function of γ'', for $\gamma' = 0, 1, 2, 3, 4, 5$. It is clear that $S(\gamma')/S(\gamma'')$ is an increasing function of γ''. However, increasing γ' significantly reduces the bound.

5.3. Experimental results

In this section, we demonstrate and discuss some experimental results.

In Table 2, we show our experimental results for task scheduling on cloud-assisted edge servers. These results are produced using the following procedure. For each $n = 20, 40, ..., 200$, we generate $N = 2000$ lists of random tasks. For each list L, we apply Algorithms 1 or 2 to get heuristic schedule length $T(L)$, calculate the lower bound R/S for the optimal schedule length $T^*(L)$ using Theorem 1, and record the ratio

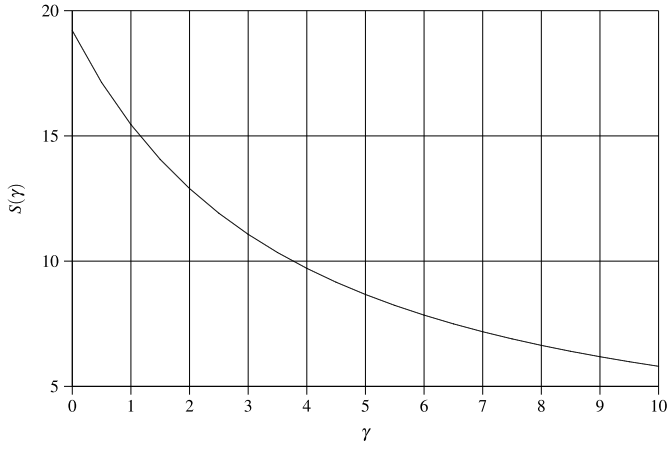


Fig. 3. $S(\gamma)$ as a decreasing function of γ.

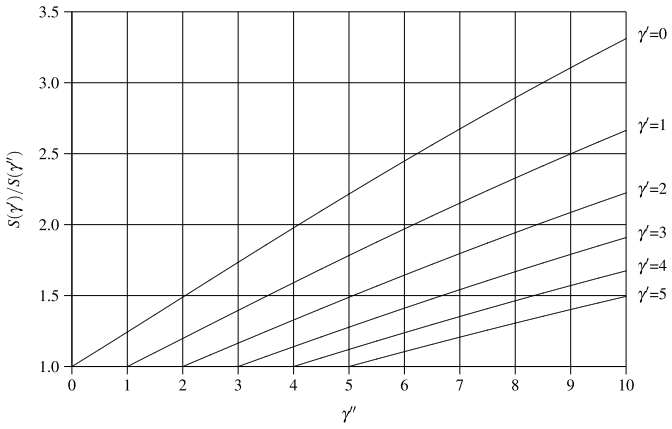


Fig. 4. $S(\gamma')/S(\gamma'')$ vs. γ''.

$T(L)/(R/S)$ as an upper bound for $T(L)/T^*(L)$. The average of the ratio is reported in the table, with 99% confidence interval $\pm 0.40712\%$. For the purpose of comparison, we adopt two server orders: $\tilde{s}_1 < \tilde{s}_2 < \tilde{s}_3 < \tilde{s}_4$ and $\tilde{s}_1 > \tilde{s}_2 > \tilde{s}_3 > \tilde{s}_4$. As a reference, the asymptotic performance bound from Theorem 3 is $S(1)/S(5) = 1.78281$.

We have the following observations.

- The schedule lengths of both Algorithms 1 or 2 are fairly close to the optimal schedule length. This means that our heuristic task scheduling algorithms are quite effective in handling a sophisticated computing environment.
- As n increases, the gap between heuristic schedule length and optimal schedule length is smaller. This means that the more the tasks, the better the performance of our heuristic task scheduling algorithms.
- Algorithm 2 consistently outperforms Algorithm 1. This means that even slight look-ahead improves the performance of a heuristic task scheduling algorithm.
- The server order does have impact on the performance. Both Algorithms 1 and 2 perform better for the order of $\tilde{s}_1 > \tilde{s}_2 > \tilde{s}_3 > \tilde{s}_4$ than the order of $\tilde{s}_1 < \tilde{s}_2 < \tilde{s}_3 < \tilde{s}_4$.
- All data in the table are less than 1.78281. This means that there is room to tighten the asymptotic performance bound in Theorem 3, either raising the lower bound or reducing the upper bound through more thorough analysis.

6. Energy-constrained scheduling

In this section, we design and analyze heuristic algorithms for energy-constrained task scheduling on cloud-assisted edge servers.

6.1. Model and problem

In addition to the models in Section 2.1, we also need power consumption models for both computation and communication. As mentioned earlier, we only consider energy consumption of the mobile device. All power consumptions are measured by Watts, and all energy consumptions are measured by Joule.

The power consumption P for computation of the mobile device is modeled by

$$P = P_d + P_s = \xi s_d^{\alpha} + P_s, \tag{30}$$

where $P_d = \xi s_d^{\alpha}$ is the dynamic component of power consumption, P_s is the static component of power consumption, and ξ, α are technology-dependent constants [20,21].

The power consumption for communication (i.e., the transmission power) between the mobile device and ES_j is modeled by

$$P_{t,j} = \frac{2^{\tilde{s}_j/w_j} - 1}{\beta_j}, \tag{31}$$

where

$$\tilde{s}_j = w_j \log_2(1 + \beta_j P_{t,j}) \tag{32}$$

is the wireless communication speed between the mobile device and ES_j, w_j is the communication bandwidth (measured by Mbps), and β_j is a quantity determined by the communication channel [20,21].

For a given schedule, the total energy consumption of the mobile device is

$$E = PT_d + \sum_{j=1}^{m} P_{t,j}\phi_j T'_j, \tag{33}$$

which is actually

$$E = (\xi s_d^{\alpha} + P_s)\frac{R_d}{s_d} + \sum_{j=1}^{m}\left(\frac{2^{\tilde{s}_j/w_j} - 1}{\beta_j}\right)\frac{D_j}{\tilde{s}_j}. \tag{34}$$

We are now ready to define our second combinatorial optimization problem.

Problem 2. Energy-Constrained Task Scheduling on Cloud-Assisted Edge Servers.

Input: $L = (t_1, t_2, ..., t_n)$, ES_1, ES_2, ..., ES_m, and $\hat{E}$.

Output: A schedule of L: $L_d, L_1, L_2, ..., L_m$, computation speed s_d, and communication speeds $\tilde{s}_j$ for all $1 \le j \le m$, such that T is minimized and that $E = \hat{E}$.

The problem is NP-hard even for one regular edge server [21].

6.2. Heuristic algorithms

Our energy-constrained task scheduling problem on cloud-assisted edge servers includes two subproblems. The first subproblem is to decide a schedule of L. The second subproblem is to determine computation and communication speeds of the mobile device. We notice that any heuristic algorithm H for task scheduling on cloud-assisted edge servers (e.g., those in Section 3) can be extended to energy-constrained task scheduling on cloud-assisted edge servers. The extended algorithm is called H^*.

For the first subproblem, algorithm H^* finds a schedule of L by using the heuristic algorithm H, with some reasonable setting of the computation and communication speeds of the mobile device. The purpose is to get a partition of L into $(m+1)$ sublists: $L_d, L_1, L_2, ..., L_m$.

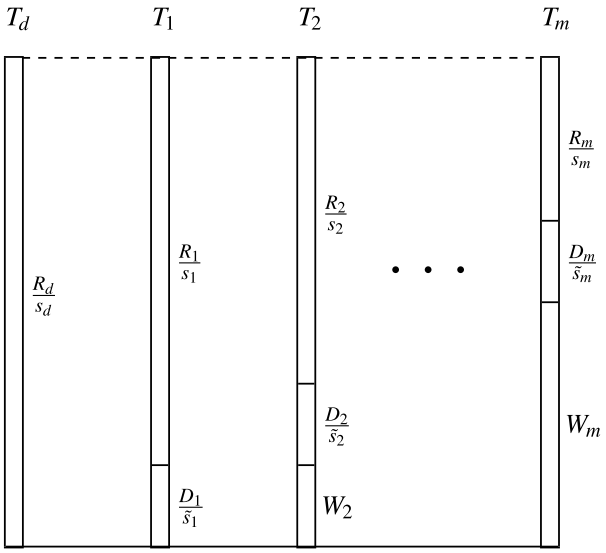


Fig. 5. Determination of computation and communication speeds.

For the second subproblem, algorithm H^* applies the following strategy to set the computation and communication speeds of the mobile device. For a given schedule of L: $L_d, L_1, L_2, ..., L_m$, s_d and $\tilde{s}_j$ are determined in such a way that the mobile device and all edge servers have the same total time (see Fig. 5), i.e., $T_d = T_j = \max\{T_{j,e}, T_{j,c}\} = T$, and that $E = \hat{E}$ by appropriate adaptation of s_d and $\tilde{s}_j$.

From $R_d/s_d = T$, we get $s_d = R_d/T$. Let

$$T_j'' = \max\left\{\frac{R_{j,e}}{s_{j,e}}, \frac{D_{j,c}}{\tilde{s}_{j,c}} + \frac{R_{j,c}}{s_{j,c}}\right\}$$

be the computation time of ES_j, which is a known quantity. From

$$\frac{D_j}{\tilde{s}_j} = T_{j-1}'' - T_j'',$$

where $T_0'' = T$, we get

$$\tilde{s}_j = \frac{D_j}{T_{j-1}'' - T_j''}.$$

Therefore, we have the following equation:

$$E = \xi\frac{R_d^\alpha}{T^{\alpha-1}} + P_s T + \sum_{j=1}^{m}\left(\frac{2^{(D_j/w_j)/(T_{j-1}''-T_j'')} - 1}{\beta_j}\right)(T_{j-1}'' - T_j'') = \hat{E}.$$

Since all T_j'' are fixed, we get

$$\begin{aligned}\xi\frac{R_d^\alpha}{T^{\alpha-1}} + P_s T + \left(\frac{2^{(D_1/w_1)/(T-T_1'')} - 1}{\beta_1}\right)(T - T_1'') \\ = \hat{E} - \sum_{j=2}^{m}\left(\frac{2^{(D_j/w_j)/(T_{j-1}''-T_j'')} - 1}{\beta_j}\right)(T_{j-1}'' - T_j'').\end{aligned} \tag{35}$$

The last equation means that for a given schedule L, we basically can only adjust the computation time of the mobile device and the wireless communication time of ES_1.

The above equation for T can be solved numerically by using the standard bisection search method, based on the fact that the left-hand side is a decreasing function of T. The initial search interval is $[lb, ub]$. For lb, since $T > T_1''$, we can set $lb = T_1''$. For ub, since [20]

$$s_d \geq \left(\frac{P_s}{\xi(\alpha-1)}\right)^{1/\alpha},$$

we can set

$$ub = R_d\left(\frac{\xi(\alpha-1)}{P_s}\right)^{1/\alpha}.$$

The above analysis implies that in order to achieve identical total time, two conditions should be satisfied. First, we must have $T_1'' > T_2'' > \cdots > T_m''$, i.e., there is room for balancing. If $T_j'' \leq T_{j+1}''$, it is impossible to achieve $T_j = T_{j+1}$, since the computation time of ES_{j+1} is too long and even longer after the wireless communication time is added. Second, the right-hand side of Eq. (35) must be positive, i.e., there is enough energy budget. As a matter of fact, this condition can be made stronger, i.e., the right-hand side of Eq. (35) must be at least the amount of energy consumption when $T = ub$.

The time complexity of algorithm H^* is the time of H plus $O(\log(\Delta/\epsilon))$, where Δ is the length of the initial search interval and ϵ is the accuracy requirement.

6.3. Lower bound

To derive a lower bound for the optimal solution, we notice that the total energy consumption of the mobile device is

$$E = (\xi s_d^\alpha + P_s)T_d + \sum_{j=1}^{m}\left(\frac{2^{\tilde{s}_j/w_j} - 1}{\beta_j}\right)\phi_j T_j', \tag{36}$$

where the wireless communication time of ES_j is $D_j/\tilde{s}_j = \phi_j T_j'$.

Based on the analysis in Section 4.2, we have $T_d = T$, and $T_j' = Q_j T$, where $T = R/S$. Hence, we have

$$E = (\xi s_d^\alpha + P_s)T + \sum_{j=1}^{m}\left(\frac{2^{\tilde{s}_j/w_j} - 1}{\beta_j}\right)\phi_j Q_j T = \hat{E}. \tag{37}$$

Furthermore, $\gamma = \gamma'$ (see Section 4.4).

To minimize T, we need to maximize the aggregated execution speed S, which is treated as a function of $s_d, \tilde{s}_1, ..., \tilde{s}_m$:

$$S(s_d, \tilde{s}_1, ..., \tilde{s}_m) = s_d + \sum_{j=1}^{m} Q_j \tilde{s}_j, \tag{38}$$

subject to the constraint $E = \hat{E}$, i.e.,

$$F(s_d, \tilde{s}_1, ..., \tilde{s}_m) = \left(\xi s_d^\alpha + P_s + \sum_{j=1}^{m}\left(\frac{2^{\tilde{s}_j/w_j} - 1}{\beta_j}\right)\phi_j Q_j\right) - \frac{\hat{E}}{R}S = 0. \tag{39}$$

To this end, we require

$$\nabla S(s_d, \tilde{s}_1, ..., \tilde{s}_m) = \lambda \nabla F(s_d, \tilde{s}_1, ..., \tilde{s}_m),$$

where λ is a Lagrange multiplier. Expanding the above equation, we obtain

$$\frac{\partial S(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial s_d} = \lambda\frac{\partial F(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial s_d},$$

and

$$\frac{\partial S(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial \tilde{s}_j} = \lambda\frac{\partial F(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial \tilde{s}_j},$$

for all $1 \leq j \leq m$.

Since

$$\frac{\partial S(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial s_d} = 1,$$

and

$$\frac{\partial F(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial s_d} = \xi\alpha s_d^{\alpha-1} - \frac{\hat{E}}{R},$$

we get

$$1 = \lambda\left(\xi\alpha s_d^{\alpha-1} - \frac{\hat{E}}{R}\right),$$

which implies that

$$s_d = \left(\frac{1}{\xi\alpha}\left(\frac{1}{\lambda} + \frac{\hat{E}}{R}\right)\right)^{1/(\alpha-1)}.$$

Notice that

$$\frac{\partial \bar{s}_j}{\partial \tilde{s}_j} = \frac{\gamma s_j^2}{(\gamma s_j + \tilde{s}_j)^2},$$

and

$$\frac{\partial \phi_j}{\partial \tilde{s}_j} = -\frac{\gamma s_j}{(\gamma s_j + \tilde{s}_j)^2},$$

and

$$\frac{\partial Q_{j'}}{\partial \tilde{s}_j} = -\frac{Q_{j'}}{1-\phi_j} \cdot \frac{\partial \phi_j}{\partial \tilde{s}_j}, \quad j' > j.$$

It is clear that

$$\frac{\partial S(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial \tilde{s}_j} = Q_j \frac{\partial \bar{s}_j}{\partial \tilde{s}_j} + \sum_{j'>j} \bar{s}_{j'} \frac{\partial Q_{j'}}{\partial \tilde{s}_j}.$$

Furthermore,

$$\begin{aligned}\frac{\partial F(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial \tilde{s}_j} &= Q_j\left(\frac{2^{\bar{s}_j/w_j}\ln 2}{\beta_j w_j}\phi_j + \frac{2^{\bar{s}_j/w_j}-1}{\beta_j} \cdot \frac{\partial \phi_j}{\partial \tilde{s}_j}\right) \\ &\quad + \sum_{j'>j}\left(\frac{2^{\bar{s}_{j'}/w_{j'}}-1}{\beta_{j'}}\right)\phi_{j'}\frac{\partial Q_{j'}}{\partial \tilde{s}_j} \\ &\quad - \frac{\hat{E}}{R} \cdot \frac{\partial S(s_d, \tilde{s}_1, ..., \tilde{s}_m)}{\partial \tilde{s}_j}.\end{aligned}$$

Thus, we have

$$\begin{aligned}&\left(1 + \lambda\frac{\hat{E}}{R}\right)\left(Q_j\frac{\partial \bar{s}_j}{\partial \tilde{s}_j} + \sum_{j'>j}\bar{s}_{j'}\frac{\partial Q_{j'}}{\partial \tilde{s}_j}\right) \\ &= \lambda\left(Q_j\left(\frac{2^{\bar{s}_j/w_j}\ln 2}{\beta_j w_j}\phi_j + \frac{2^{\bar{s}_j/w_j}-1}{\beta_j} \cdot \frac{\partial \phi_j}{\partial \tilde{s}_j}\right)\right. \\ &\quad \left. + \sum_{j'>j}\left(\frac{2^{\bar{s}_{j'}/w_{j'}}-1}{\beta_{j'}}\right)\phi_{j'}\frac{\partial Q_{j'}}{\partial \tilde{s}_j}\right),\end{aligned} \tag{40}$$

for all $1 \le j \le m$.

We now have a nonlinear system of $(m+1)$ equations (i.e., Eqs. (39) and (40)) with $(m+1)$ unknowns (i.e., $\tilde{s}_1, ..., \tilde{s}_m$ and λ). Once these equations are solved, we can calculate the maximized S and a lower bound for the optimal solution, which is R/S.

6.4. Experimental results

Both Algorithms 1 and 2 in Section 3 have been extended to Algorithms 1* and 2* for energy-constrained task scheduling on cloud-assisted edge servers.

We adopt the same parameter setting in Section 5. The parameters for our computation power consumption model are $\xi = 0.1$, $\alpha = 2.0$, and $P_s = 0.05$ Watts. The parameters for our communication power consumption model are $w_j = 30$ Mbps and $\beta_j = 2.0$ Watts^{-1}.

Table 3
Experimental results for energy-constrained scheduling on cloud-assisted edge servers.

	$\tilde{s}_1 < \tilde{s}_2 < \tilde{s}_3 < \tilde{s}_4$		$\tilde{s}_1 > \tilde{s}_2 > \tilde{s}_3 > \tilde{s}_4$	
n	Algorithm 1*	Algorithm 2*	Algorithm 1*	Algorithm 2*
20	1.87961	1.72850	1.82672	1.72529
40	1.86006	1.75466	1.76430	1.68097
60	1.84079	1.74692	1.69061	1.63369
80	1.78364	1.70234	1.63551	1.59209
100	1.73387	1.66032	1.61646	1.57966
120	1.69448	1.63382	1.59419	1.56371
140	1.67791	1.62366	1.58787	1.55916
160	1.66602	1.61724	1.58196	1.55435
180	1.65570	1.60953	1.58012	1.55403
200	1.64676	1.60234	1.57618	1.55030

In Table 3, we show our experimental results for energy-constrained task scheduling on cloud-assisted edge servers. These results are produced using the following procedure. The energy constraint is set as $\hat{E} = 0.2n$ Joules. The nonlinear system of equations is solved by using Python scipy.optimize.fsolve, with $R = 2.5n$ GI. For each $n = 20, 40, ..., 200$, we generate $N = 5000$ lists of random tasks. For each list L, we apply Algorithms 1* and 2* to get heuristic schedule length $T(L)$, use the lower bound R/S for the optimal schedule length $T^*(L)$, and record the ratio $T(L)/(R/S)$ as an upper bound for $T(L)/T^*(L)$. The average of the ratio is reported in the table, with 99% confidence interval $\pm 1.02984\%$. For the purpose of comparison, we adopt two server orders: $\tilde{s}_1 < \tilde{s}_2 < \tilde{s}_3 < \tilde{s}_4$ and $\tilde{s}_1 > \tilde{s}_2 > \tilde{s}_3 > \tilde{s}_4$.

Since $\hat{E}/R = 0.08$ is fixed, our nonlinear system of equations is identical for all n, thus only needs to be solved once. The computation and communication speeds are $s_d = 1.61601$ Bips, and

$$\begin{aligned}&\tilde{s}_1 = 139.45898, \quad s_1 = 5.87563, \quad \bar{s}_1 = 5.63809,\\ &\tilde{s}_2 = 127.89733, \quad s_2 = 6.06154, \quad \bar{s}_2 = 5.78726,\\ &\tilde{s}_3 = 107.73416, \quad s_3 = 2.70000, \quad \bar{s}_3 = 2.63399,\\ &\tilde{s}_4 = 90.35365, \quad s_4 = 2.80000, \quad \bar{s}_4 = 2.71584,\end{aligned}$$

$$\begin{aligned}&\phi_1 = 0.04043, \quad Q_1 = 1.00000;\\ &\phi_2 = 0.04525, \quad Q_2 = 0.95957;\\ &\phi_3 = 0.02445, \quad Q_3 = 0.91615;\\ &\phi_4 = 0.03006, \quad Q_4 = 0.89375.\end{aligned}$$

The aggregated execution speed is $S = 17.64781$ Bips, and the lower bound is $R/S = 0.14166n$ seconds.

We have the following observations.

- The schedule lengths of both Algorithms 1* and 2* are reasonably close to the optimal schedule length. This means that our heuristic algorithms are quite effective in handling energy-constrained task scheduling in a sophisticated computing environment.
- As n increases, the gap between heuristic schedule length and optimal schedule length is smaller. This means that the more the tasks, the better the performance of our heuristic task scheduling algorithms.
- Algorithm 2* consistently outperforms Algorithm 1*. This means that even slight look-ahead improves the performance of a heuristic task scheduling algorithm.
- The server order does have impact on the performance. Both Algorithms 1* and 2* perform better for the order of $\tilde{s}_1 > \tilde{s}_2 > \tilde{s}_3 > \tilde{s}_4$ than the order of $\tilde{s}_1 < \tilde{s}_2 < \tilde{s}_3 < \tilde{s}_4$.

7. Comments on related work

Task scheduling and offloading in device-edge-cloud cooperative computing have two main considerations.

The first consideration is performance optimization. In the current studies, many researchers have attempted to minimize the summation of task execution times (see, e.g., [7,10,23,26]). Actually, this is not combinatorial optimization and is quite different from traditional scheduling theory, where the makespan is usually the performance metrics to be minimized. This paper is the first combinatorial optimization study in the literature which optimizes the makespan of task scheduling in mobile edge computing with multiple cloud-assisted edge servers.

The second consideration is performance-cost tradeoff. In the existing literature, many researchers have attempted to minimize a weighted sum of execution time and energy consumption of all tasks (see, e.g., [6,17,25,29]). Such an approach is again not combinatorial optimization. In addition, the approach has inherent technical flaw and makes little sense, because execution time and energy consumption have totally different measures (i.e., second and Joule). This paper is the first combinatorial optimization study in the literature which optimizes the makespan of task scheduling with energy constraint in mobile edge computing with multiple cloud-assisted edge servers.

It is worth to mention that some researchers have considered energy minimization with performance guarantee [13,15,16,32].

8. Concluding remarks

We have treated task scheduling on cloud-assisted edge servers as a combinatorial optimization problem to minimize the schedule length, with or without energy constraint. We have developed heuristic algorithms that take into consideration the heterogeneity of computation and communication speeds, the sophistication of wireless and wired communication mechanisms, and the order of edge servers. The most important feature of our study is to compare the performance of our heuristic algorithms with the optimal algorithm, both analytically and experimentally. In doing so, we have adopted three key techniques, i.e., communication unification, effective speed, and virtual tasks.

Further research can be conducted for more sophisticated application structures, more challenging power allocations strategies, and different task scheduling frameworks. First, we may consider scheduling dependent (i.e., precedence constrained) tasks on cloud-assisted edge servers with or without energy constraint. Second, for dependent tasks, it is possible that each task has its own communication speed, which results in more efficient power allocation and shorter schedule length. Third, it is worth to investigate overlapping and interleaving of computations and communications (i.e., a sublist of tasks are not sent to a server altogether, but in smaller batches), which may yield shorter waiting times. In all the above cases and their combinations, it will be substantially more difficult to design efficient and effective heuristic algorithms and to analyze the optimal solutions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The author would like to thank the anonymous reviewers for their constructive comments on the manuscript.

Appendix A. Notations and definitions

Notation	Definition
L	$=(t_1,t_2,...,t_n)$, a list of independent tasks
n	the number of tasks
t_i	$=(d_i,r_i)$, a task
d_i	the amount of communication of t_i
r_i	the amount of computation of t_i
γ_i	$=d_i/r_i$, the amount of communication per unit of computation
γ'	$=\min\{\gamma_1,\gamma_2,...,\gamma_n\}$
γ''	$=\max\{\gamma_1,\gamma_2,...,\gamma_n\}$
L_d	the sublist of task executed on the mobile device
L_j	the sublist of task executed on ES_j
$L_{j,e}$	the sublist of task executed on ES_j of CES_j
$L_{j,c}$	the sublist of task executed on CS_j of CES_j
D_k	the total amount of communication of tasks in L_k, $k=d,j,(j,e),(j,c)$
R_k	the total amount of computation of tasks in L_k, $k=d,j,(j,e),(j,c)$
τ_i	the execution time of task t_i
τ^*	the longest execution time of any task anywhere
s_d	the computation speed of the mobile device
s_j	the computation speed of RES_j, or the effective computation speed of CES_j
$s_{j,e}$	the computation speed of ES_j of CES_j
$s_{j,c}$	the computation speed of CS_j of CES_j
$\bar{s}_j$	the wireless communication speed between the mobile device and ES_j
$\bar{s}_{j,c}$	the wired communication speed between ES_j and CS_j
$\bar{s}_{j,c}$	the effective computation speed of CS_j
$\bar{s}_j$	the effective execution speed of ES_j
T_d	the total time of the mobile device
T_j	$=W_j+T'_j$, the total time of RES_j, or the total time of $\mathrm{CES}_j = \max\{T_{j,e},T_{j,c}\}$
T	$=\max\{T_d,T_1,T_2,...,T_m\}$, the schedule length
$T_{j,e}$	the total time of ES_j of CES_j
$T_{j,c}$	the total time of CS_j of CES_j
W_j	the waiting time of ES_j
T'_j	the execution time of ES_j
T''_j	the computation time of ES_j
ϕ_j	the percentage of the communication time in the execution time of ES_j
Q_j	$=\prod_{j'<j}(1-\phi_{j'})$, the percentage of the execution time T'_j in the total time T_j of ES_j
S	the aggregated execution speed of the device-edge-cloud collaborative computing system
$T^*(L)$	the optimal schedule length
$T(L)$	the heuristic schedule length
P	the power consumption for computation
P_d	the dynamic component of power consumption
P_s	the static component of power consumption
ξ,α	parameters of the power consumption model for computation
$P_{t,j}$	the power consumption for communication (i.e., the transmission power)
w_j	the communication bandwidth
β_j	communication channel property
E	the total energy consumption of the mobile device
$\hat{E}$	energy constraint

References

[1] F. Bu, Q. Zhang, L.T. Yang, H. Yu, An edge-cloud-aided high-order possibilistic c-means algorithm for big data clustering, IEEE Trans. Fuzzy Syst. 28 (12) (2020) 3100–3109.

[2] Z. Cao, H. Zhang, B. Liu, B. Sheng, Revenue sharing in edge-cloud systems: a game-theoretic perspective, Comput. Netw. 176 (2020) 107286.

[3] A. Carroll, G. Heiser, An analysis of power consumption in a smartphone, in: Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference, 23 June, 2010.

[4] L. Chunlin, H. Sun, C. Yi, Y. Luo, Edge cloud resource expansion and shrinkage based on workload for minimizing the cost, Future Gener. Comput. Syst. 101 (2019) 327–340.

[5] C. Ding, A. Zhou, Y. Liu, R.N. Chang, C.-H. Hsu, S. Wang, A cloud-edge collaboration framework for cognitive service, IEEE Trans. Cloud Comput. 10 (3) (2022) 1489–1499.

[6] Y. Ding, K. Li, C. Liu, K. Li, A potential game theoretic approach to computation offloading strategy optimization in end-edge-cloud computing, IEEE Trans. Parallel Distrib. Syst. 33 (6) (2022) 1503–1519.

[7] M. Du, Y. Wang, K. Ye, C. Xu, Algorithmics of cost-driven computation offloading in the edge-cloud environment, IEEE Trans. Comput. 69 (10) (2020) 1519–1532.

[8] R. Du, Y. Liu, L. Liu, W. Du, A lightweight heterogeneous network clustering algorithm based on edge computing for 5G, Wirel. Netw. 26 (2020) 1631–1641.

[9] T.L. Duc, R.G. Leiva, P. Casari, P.-O. Östberg, Machine learning methods for reliable resource provisioning in edge-cloud computing: a survey, ACM Comput. Surv. 52 (5) (2019) 94.

[10] R. Fantacci, B. Picano, A matching game with discard policy for virtual machines placement in hybrid cloud-edge architecture for industrial IoT systems, IEEE Trans. Ind. Inform. 16 (11) (2020) 7046–7055.

[11] J. Feng, L. Liu, Q. Pei, K. Li, Min-max cost optimization for efficient hierarchical federated learning in wireless edge networks, IEEE Trans. Parallel Distrib. Syst. 33 (11) (2022) 2687–2700.

[12] R.L. Graham, Bounds on multiprocessing timing anomalies, SIAM J. Appl. Math. 17 (2) (1969) 416–429.

[13] M. Guo, L. Li, Q. Guan, Energy-efficient and delay-guaranteed workload allocation in IoT-edge-cloud computing systems, IEEE Access 7 (2019) 78685–78697.

[14] Y. Hao, Y. Jiang, T. Chen, D. Cao, M. Chen, iTaskOffloading: intelligent task offloading for a cloud-edge collaborative system, IEEE Netw. 33 (5) (2019) 82–88.

[15] Z. He, Y. Xu, D. Liu, W. Zhou, K. Li, Energy-efficient computation offloading strategy with task priority in cloud assisted mobile edge computing, Future Gener. Comput. Syst. 148 (2023) 298–313.

[16] Z. He, Y. Xu, M. Zhao, W. Zhou, K. Li, Priority-based offloading optimization in cloud-edge collaborative computing, IEEE Trans. Serv. Comput. (2023), in press.

[17] M. Huang, W. Liu, T. Wang, A. Liu, S. Zhang, A cloud-MEC collaborative task offloading scheme with service orchestration, IEEE Int. Things J. 7 (7) (2020) 5792–5805.

[18] Y. Huang, F. Wang, F. Wang, J. Liu, DeePar: a hybrid device-edge-cloud execution framework for mobile deep learning applications, in: IEEE Conference on Computer Communications Workshops, Paris, France, 29 April–2 May, 2019.

[19] C. Li, H. Sun, H. Tang, Y. Luo, Adaptive resource allocation based on the billing granularity in edge-cloud architecture, Comput. Commun. 145 (2019) 29–42.

[20] K. Li, Heuristic computation offloading algorithms for mobile users in fog computing, ACM Trans. Embed. Comput. Syst. 20 (2) (2021) 11, 28 pp.

[21] K. Li, Design and analysis of heuristic algorithms for energy-constrained task scheduling with device-edge-cloud fusion, IEEE Trans. Sustain. Comput. 8 (2) (2023) 208–221.

[22] F.P.-C. Lin, Z. Tsai, Hierarchical edge-cloud SDN controller system with optimal adaptive resource allocation for load-balancing, IEEE Syst. J. 14 (1) (2020) 265–276.

[23] B. Liu, W. Zhang, W. Chen, H. Huang, S. Guo, Online computation offloading and traffic routing for UAV swarms in edge-cloud computing, IEEE Trans. Veh. Technol. 69 (8) (2020) 8777–8791.

[24] R. Mahmud, S.N. Srirama, K. Ramamohanarao, R. Buyya, Profit-aware application placement for integrated fog-cloud computing environments, J. Parallel Distrib. Comput. 135 (2020) 177–190.

[25] K. Peng, H. Huang, S. Wan, V.C.M. Leung, End-edge-cloud collaborative computation offloading for multiple mobile users in heterogeneous edge-server environment, Wireless Networks, published online: 10 June 2020, https://doi.org/10.1007/s11276-020-02385-1.

[26] J. Ren, G. Yu, Y. He, G.Y. Li, Collaborative cloud and edge computing for latency minimization, IEEE Trans. Veh. Technol. 68 (5) (2019) 5031–5044.

[27] J. Ren, D. Zhang, S. He, Y. Zhang, T. Li, A survey on end-edge-cloud orchestrated network computing paradigms: transparent computing, mobile edge computing, fog computing, and cloudlet, ACM Comput. Surv. 52 (6) (2019) 125.

[28] F. Song, M. Zhu, Y. Zhou, I. You, H. Zhang, Smart collaborative tracking for ubiquitous power IoT in edge-cloud interplay domain, IEEE Int. Things J. 7 (7) (2020) 6046–6055.

[29] C. Sun, H. Li, X. Li, J. Wen, Q. Xiong, X. Wang, V.C.M. Leung, Task offloading for end-edge-cloud orchestrated computing in mobile networks, in: IEEE Wireless Communications and Networking Conference, Seoul, South Korea, 25-28 May, 2020.

[30] S. Wang, S. Yang, C. Zhao, SurveilEdge: real-time video query based on collaborative cloud-edge deep learning, in: IEEE Conference on Computer Communications, Toronto, ON, Canada, 6-9 July, 2020.

[31] T. Wang, Y. Lu, J. Wang, H.-N. Dai, X. Zheng, W. Jia, EIHDP: edge-intelligent hierarchical dynamic pricing based on cloud-edge-client collaboration for IoT systems, IEEE Trans. Comput. 70 (8) (2021) 1285–1298.

[32] L. Yin, J. Zhou, J. Sun, Z. Gu, K. Li, ECFA: an efficient convergent firefly algorithm for solving task scheduling problems in cloud-edge computing, IEEE Trans. Serv. Comput. 16 (5) (2023) 3280–3293.

[33] A. Younis, B. Qiu, D. Pompili, Latency-aware hybrid edge cloud framework for mobile augmented reality applications, in: 17th IEEE International Conference on Sensing, Communication, and Networking, Como, Italy, 22–25 June, 2020.

[34] Y. Zhang, X. Lan, J. Ren, L. Cai, Efficient computing resource sharing for mobile edge-cloud computing networks, IEEE/ACM Trans. Netw. 28 (3) (2020) 1227–1240.

Keqin Li received the B.S. degree in computer science from Tsinghua University in 1985 and the Ph.D. degree in computer science from the University of Houston in 1990. He is currently a SUNY Distinguished Professor with the State University of New York and a National Distinguished Professor with Hunan University (China). He has authored or coauthored more than 950 journal articles, book chapters, and refereed conference papers. He received several best paper awards from international conferences including PDPTA-1996, NAECON-1997, IPDPS-2000, ISPA-2016, NPC-2019, ISPA-2019, CPSCom-2022. He holds nearly 70 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top five most influential scientists in parallel and distributed computing in terms of both single-year impact and career-long impact based on a composite indicator of Scopus citation database. He was a 2017 recipient of the Albert Nelson Marquis Lifetime Achievement Award for being listed in Marquis Who's Who in Science and Engineering, Who's Who in America, Who's Who in the World, and Who's Who in American Education for more than twenty consecutive years. He received the Distinguished Alumnus Award of the Computer Science Department at the University of Houston in 2018. He received the IEEE TCCLD Research Impact Award from the IEEE CS Technical Committee on Cloud Computing in 2022 and the IEEE TCSVC Research Innovation Award from the IEEE CS Technical Community on Services Computing in 2023. He is a Member of SUNY Distinguished Academy. He is an AAAS Fellow, an IEEE Fellow, and an AAIA Fellow. He is a Member of Academia Europaea (Academician of the Academy of Europe).