Real-Time Task Scheduling in Fog-Cloud Computing Framework for IoT Applications: A Fuzzy Logic based Approach

Hala S. Ali[‡], Rashmi Ranjan Rout[‡], Priyanka Parimi[‡], Sajal K. Das[§]
[‡]Department of Computer Science and Engineering, National Institute of Technology Warangal, Telangana, 506004, India.
Email: {halaali@student.nitw.ac.in, rashrr@nitw.ac.in, priyankap@student.nitw.ac.in}
[§]Department of Computer Science, Missouri University of Science and Technology Rolla, MO 65409, USA
Email: sdas@mst.edu

2021 International Conference on COMmunication Systems & NETworkS (COMSNETS) | 978-1-7281-9127-0/20/$31.00 ©2021 IEEE | DOI: 10.1109/COMSNETS51098.2021.9352931

***Abstract*—As an extension of the cloud, a fog computing environment facilitates the deployment of Internet of Things (IoT) applications by shifting the computing, storage and networking services closer to the IoT devices, thus satisfying the delay and response time requirements. This paper aims to improve the overall task execution efficiency of IoT applications by appropriately selecting customized real-time tasks for execution at the fog layer. Specifically, we propose a fuzzy logic based task scheduling algorithm to divide the tasks between the fog and cloud layers in a fog-cloud computing framework. The algorithm selects appropriate processing units to execute the submitted tasks in the fog layer with heterogeneous resources, by exploiting the task requirements (e.g., computation, storage, bandwidth) and their constraints (e.g., deadline, data size). Simulation experiments demonstrate the efficacy of the proposed algorithm and its superior performance as compared to other existing algorithms in terms of success ratio of the tasks, makespan, average turnaround time, and delay rate.**

***Index Terms*—Cloud computing, fog computing, fuzzy logic, IoT, real-time task scheduling**

I. Introduction

Internet of Things (IoT) applications typically utilize the cloud computing platform because of its virtually abundant computing power and data storage capabilities [1], [2]. With the rapid growth of IoT and real-time applications, cloud computing is not suitable for the time-sensitive IoT applications due to high communication delay between the cloud data centers and IoT devices [3]. Fog Computing aims to reduce the network delay by providing the required resources at the network edge closer to the end devices [4] [5], thus helping reduce latency, improve quality of service, and support mobility [6]. However, the fog resources are limited and cannot process locally the massive amount of data generated by the IoT devices. Therefore, effectively selecting the processing of tasks between the cloud server and fog server is important to satisfy the ever-increasing and customized demands of IoT applications. One of the major challenges addressed in this paper is to decide whether the requested tasks will be processed in the fog layer or forwarded to the cloud layer in order to improve the overall efficiency of time-sensitive IoT applications.

In the literature, several decision algorithms have been proposed for dividing the tasks between the cloud and fog based on parameters like execution time and deadline of the tasks [7], task length [8], security and deadline of the tasks [9], etc. However, these works do not adequately address the resource requirements of the tasks taking into account the limited resource availability of the fog. Further, the resource demands of the tasks cannot be determined accurately at the time of task submission [10]. In the fog-cloud environment, the dynamic requirements of the IoT tasks can be efficiently processed by a fuzzy mechanism, which is the focus of this work. Fuzzy logic has been the preferred method in many domains because its computational complexity and processing power requirements are low compared to other decision making methods [11] [12] [13]. The changing features of the tasks as well as the dynamic changes of the fog-cloud environment lead to new challenges for task scheduling. This motivates us to propose a fuzzy logic based decision algorithm to distribute the tasks among the cloud and fog layers based on the time constraints and resource requirements of the tasks, the network latency to the cloud, and the heterogeneous resource configurations of the fog nodes.

Indeed, the diverse requirements of tasks and different resource capabilities of the fog nodes affect the task processing times. Therefore, assigning a task to an appropriate processing node in the fog layer in a heterogeneous fog-cloud environment is crucial. In [8], an edge node attempts to execute a task locally and submits it to other edge nodes or cloud when it fails. However, in a fog environment (say edge node) with heterogeneous virtual machines, an appropriate virtual machine selection criteria (for tasks) is an important requirement to improve the overall efficiency of task execution. Moreover, a *fog broker* is used in the fog layer to manage the fog resources and schedule the tasks in fog nodes, depending on such parameters as the resource demand and availability [14], and delay tolerance of the task [15].

To this end, in this work, we propose a real-time task scheduling algorithm at the fog broker by considering the deadline and resource requirements of tasks in addition to the resource availability and workload of the fog nodes. In particular, our proposed algorithm creates an efficient communication

Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.

and task execution framework for fog computing to support IoT based customized tasks where important considerations are time constraints of the tasks, available resources in the fog layer, and the latency between the cloud and fog layers.

The major contributions of this paper are summarized as:

1) Design a novel fuzzy logic based decision algorithm to distribute tasks between the fog and cloud layers in a fog-cloud framework.
2) Design an improved task scheduling algorithm for the fog broker to schedule the tasks among heterogeneous fog nodes.
3) Simulate the proposed algorithms (using *iFogSim* simulator) and compare with other existing approaches using parameters like *Makespan*, *Delay rate*, *Success ratio*, *Average Turnaround Time* and *Average Processing Time*.

The rest of the paper is organized as follows. Section II summarizes the related work. The fog-cloud computing model is discussed in Section III along with the decision making and task scheduling problems. Section IV presents the proposed fuzzy logic based decision algorithm and the task scheduling algorithm for the fog broker. Simulation results are discussed in Section V. Finally, Section VI concludes the paper.

II. Related Work

Several studies have been done on task scheduling and resource management in fog-cloud computing. In [16], a task scheduling algorithm is proposed in fog-cloud environment by considering the task priority level, deadline, and resources available in the fog layer. The proposed algorithm minimizes the task execution time and cost through resource allocation and load balance. The authors in [15] proposed the Time-Cost aware Scheduling algorithm to optimize a trade-off between the execution time and operating cost of the tasks. In [14], the modified particle swarm optimization (MPSO) algorithm is used for task scheduling and load balancing.

An intelligent scheduling algorithm of the computing resources is proposed in [17] to fulfill the latency and deadline constraints of tasks in a smart manufacturing framework. The authors in [7] improved task completion ratio and throughput by proposing three parallel algorithms for task offloading, task buffering, and resource allocation taking into account the estimated task execution time, laxity and transmission delay to the cloud. A trade-off between the processing cost and time constraints of the task is presented in [18], where the authors proposed a genetic algorithm based scheduling in a fog-cloud architecture. However, they did not consider the resource requirements of tasks nor virtual machines and available resources in the fog layer.

The resource utilization in fog-cloud architecture have been improved in [19] where the method aims to allocate the IoT application modules to the fog and cloud nodes by considering the required resources. In [20], the authors aimed at optimizing energy consumption and the quality of service using evolutionary algorithms that assign tasks to the processing nodes.

The above works do not adequately address characteristics like resource availability at the processing node, communication overhead between the fog nodes, and communication overhead between the fog and cloud layers that can affect the task processing time. In this work, we consider the resource required by the tasks, deadline of tasks, and the load of the processing nodes along with the transmission time of the cloud. Further, the proposed algorithm executes at the fog broker and thus minimizes the communication overhead.

III. System Model and problem Statement

This section presents the fog-cloud architecture framework and the problem statement.

A. Fog-Cloud Architecture

A fog-cloud architecture framework with three layers is shown in Fig. 1. The end users submit various tasks to the fog layer through IoT devices. The fog layer comprises of a set of fog servers (having limited resources) which consist of virtual machines to provide various services at the network edge. The fog servers forward the received IoT tasks to the fog broker, which is responsible for managing the resources of the fog servers and scheduling the tasks. The fog broker being located close to the fog servers, the communication time between them is very low. The fog broker distributes the selected tasks between the fog and cloud layers, and also selects an appropriate node to execute a task in the fog layer. The cloud layer consists of powerful data centers which process the tasks forwarded by the fog broker.

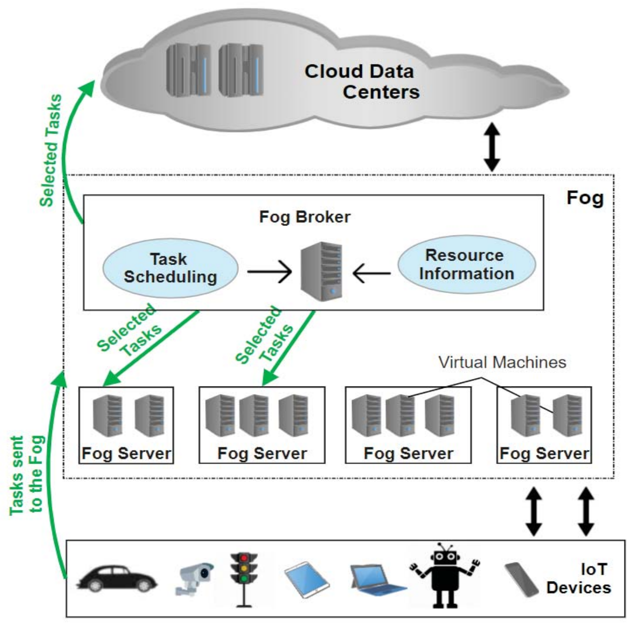


Fig. 1: Fog-Cloud Computing Architecture Framework with Task scheduling

In this work, we consider that a task received by the fog broker is independent, inseparable and non-preemptive. Let $\widehat{T}$ be the set of tasks received by the fog broker. Each task is represented as $t_i = (tid, in, ds, rs, dl, ct, st, bt)$ where tid

Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.

is the unique identifier of the task, in is the number of instructions in the task, ds is the size of the data accompanying the task, rs is the size of the task result, and dl is the deadline of the task. The terms ct, st, bt represent the CPU rate, storage and bandwidth required for the task, respectively. Each fog server has several heterogeneous virtual machines where the tasks are executed. Let V be the set of all virtual machines in the fog layer. A virtual machine is represented as $v_j = (vid, type, mc, cc, sc, bc)$ where vid is the unique identifier of the virtual machine, and $type$ indicates whether the virtual machine is standard or computational, storage, bandwidth oriented. The terms mc, cc, sc, bc represent the memory capacity, CPU rate, storage capacity and bandwidth capacity of the virtual machine, respectively. A fog server hosts the virtual machines (number and type) that can be accommodated based on the server's resource availability [19].

B. Problem Statement

We aim to minimize the total processing time of task t_i:

$$T_{proc}(t_i) = T_{trans}(t_i) + T_{exec}(t_i) + T_{queue}(t_i) + T_{rep}(t_i) \tag{1}$$

where T_{trans} is the transmission delay of the task from an IoT device to the fog layer if the task is processed at the fog layer. Otherwise, T_{trans} will be the transmission delay of the task when it is sent by the IoT device to the cloud layer for execution. It is calculated as the ratio of the data size of the task ds and the transmission rate [17]. T_{exec} is the execution time of the task by the selected virtual machine, and T_{queue} is the waiting time for the virtual machine's availability. The task's turnaround time is the summation of T_{exec} and T_{queue} (assumed to be negligible when the tasks are processed at the cloud). However, when the task is processed in a fog node, the execution time T_{exec} primarily depends on the capacity of a virtual machine v_j. Here T_{rep} is the time taken to get the processing results at the IoT devices (i.e., the results are sent from the fog or cloud layer after execution). T_{rep} is calculated as the ratio of the size of the task result rs to the transmission rate [17]. Moreover, the total processing time of the task t_i must satisfy the condition $T_{proc}(t_i) \le dl(t_i)$.

Upon receiving a task, the fog broker decides whether to execute the task in the fog or forward it to the cloud. If it is to be executed at the fog layer, the fog broker assigns the task to an appropriate virtual machine. The first step of selecting between the cloud and the fog can be expressed as a function:

$$f : \widehat{T} \to \{Cloud, Fog\} \tag{2}$$

The second step of the scheduling process is an optimization problem of selecting a virtual machine in the fog layer:

$$\begin{gathered} \min g(t_i, v_j) = T_{exec}(t_i, v_j) + T_{queue}(t_i, v_j) \\ \text{such that } T_{proc}(t_i) \le dl(t_i) \\ \text{where } t_i \in \widehat{T},\ v_j \in V,\ f(t_i) = Fog \end{gathered} \tag{3}$$

where $g(t_i, v_j)$ is the turnaround time of the task t_i assigned to a virtual machine v_j.

IV. Proposed Fuzzy Logic based Task Scheduling

This section first proposes a fuzzy logic based decision algorithm for distributing tasks between the fog and cloud layers (Step 1), and a real-time task scheduling algorithm for selecting the best virtual machine to execute the task (Step 2).

A. Fuzzy Logic based Decision Algorithm (FLDA)

The fuzzy decision algorithm decides where to process the task based on the resource requirements, the time constraints of the tasks, the maximum available resources in the fog layer, and the latency between the cloud and fog. The minimum values of the resource utilization requirements of the task, in terms CPU rate $c_{min}(t_i)$, storage $s_{min}(t_i)$ and bandwidth $b_{min}(t_i)$, if the task t_i is assigned to the fog, is determined as

$$c_{min}(t_i) = \frac{ct(t_i)}{cc_{max}v}, \quad s_{min}(t_i) = \frac{st(t_i)}{sc_{max}v}, \quad \text{and} \quad b_{min}(t_i) = \frac{bt(t_i)}{bc_{max}v} \tag{4}$$

where $cc_{max}v = \max_{v_j \in V} cc(v_j)$ is the maximum CPU rate provided in the fog layer. Similarly, $sc_{max}v = \max_{v_j \in V} sc(v_j)$ and $bc_{max}v = \max_{v_j \in V} bc(v_j)$ are respectively the maximum storage capacity and bandwidth capacity available in the fog layer. The minimum values of the resource utilization requirements (e.g., CPU, storage, bandwidth) of the task and its constraints (e.g., deadline, network latency between fog and cloud) are normalized using the min-max normalization method. The crisp values of these parameters are converted into linguistic membership levels using triangular and trapezoidal membership functions. The output is the decision variable indicating where the task is processed. Table II describes the fuzzy sets for the input and output parameters.

The membership functions for the parameters, illustrated in Fig. 2, are calculated following Equations (5) and (6) as

$$\mu(x) = \begin{cases} 0, & x > 0.5 \\ \frac{0.5-x}{0.5-0.2}, & 0.2 \le x \le 0.5 \\ 1, & x < 0.2 \end{cases} \tag{5a}$$

$$\mu(x) = \begin{cases} 0, & x \le 0.2 \\ \frac{x-0.2}{0.5-0.2}, & 0.2 < x \le 0.5 \\ \frac{0.8-x}{0.8-0.5}, & 0.5 < x < 0.8 \\ 0, & x \ge 0.8 \end{cases} \tag{5b}$$

$$\mu(x) = \begin{cases} 0, & x < 0.5 \\ \frac{x-0.5}{0.8-0.5}, & 0.5 \le x \le 0.8 \text{ ,} \\ 1, & x > 0.8 \end{cases} \tag{5c}$$

$$\mu(x) = \begin{cases} 0, & x > 0.6 \\ \frac{0.6-x}{0.6-0.4}, & 0.4 \le x \le 0.6 \\ 1, & x < 0.4 \end{cases} \quad \text{Fog Layer} \tag{6a}$$

$$\mu(x) = \begin{cases} 0, & x < 0.4 \\ \frac{x-0.4}{0.6-0.4}, & 0.4 \le x \le 0.6 \\ 1, & x > 0.6 \end{cases} \quad \text{Cloud Layer} \tag{6b}$$

Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.

TABLE I: Fuzzy Logic Rules

CPU Utilization	Storage Utilization	Bandwidth Utilization	Task Deadline	Network Latency	Task Execution at
low	Low	Low	Hard	Low	Fog Layer
low	Low	Low	Hard	Medium	Fog Layer
low	Low	Low	Hard	High	Fog Layer
low	Low	Medium	Soft	High	Fog Layer
Medium	Medium	Low	Soft	Low	Cloud Layer
Medium	Medium	Low	Soft	Medium	Fog Layer
⋮	⋮	⋮	⋮	⋮	⋮
High	Low	High	Hard	High	Fog Layer
High	High	Medium	Soft	Low	Cloud Layer
High	High	High	Hard	Low	Cloud Layer
High	High	High	Medium	High	Fog Layer

TABLE II: Fuzzy Input/Output Variables and Corresponding Fuzzy Sets

Input/output Variables	Fuzzy Sets
CPU Utilization	Low, Medium, High
Storage Utilization	Low, Medium, High
Bandwidth Utilization	Low, Medium, High
Task deadline	Hard, Medium, Soft
Network Latency	Low, Medium, High
Task Execution	Fog Layer, Cloud Layer

where x is the crisp value of the parameter and $\mu(x)$ is the membership value between 0 and 1.

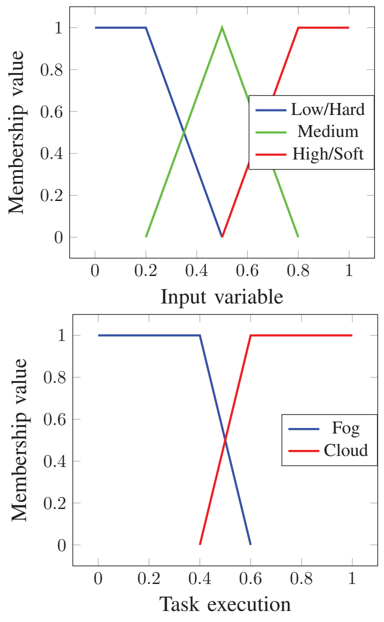


Fig. 2: Membership functions of the Input and Output fuzzy set variables

In our work, we have generated the fuzzy logic rule base of 243 rules, based on the five input variables and one output variable, using experimental data. A sample of the rule base is shown in Table I. The *Mamdani Fuzzy Inference System* has been adopted due to its simplicity [21] [22]. After the fuzzy output is obtained according to the rules, the output fuzzy sets are transformed to a single crisp value using the *Center of Gravity* (CoG) defuzzification method [23].

The *Fuzzy Logic based Decision Algorithm*, presented in Algorithm 1, consists of the predefined rule set RS and the decision threshold value D_{th}. The algorithm takes the normalized utilization of CPU, storage and bandwidth, deadline of a task and network latency between the fog and cloud layers as inputs and returns the decision on where to process the task. The membership value and the level for each input variable are determined using the triangular and trapezoidal membership functions in lines 2-4 [24]. The input membership levels are compared with each rule of the rule set and the output linguistic level is determined. The output membership value is determined as the minimum of the membership values of the input parameters. The level and value are added to the output set in lines 5-11. The output fuzzy levels are aggregated into a single fuzzy set using the *Maximum Aggregation* method [25] in line 12 and the output crisp value D is obtained by applying the CoG defuzzification method in line 13. Finally, the output crisp value D is compared with the predefined

Algorithm 1 Fuzzy Logic based Decision Algorithm (FLDA)

```
    Input $c_{min}$, $s_{min}$, $b_{min}$, $dl$, $nl$
    Output $Decision$
    Predefined $RS$, $D_{th}$
 1: Initialize output set $O = \{\}$
 2: for $x \in \{c_{min}, s_{min}, b_{min}, dl, nl\}$ do
 3:     Calculate $\mu(x)$ based on Eq. (5)
 4: end for
 5: for each rule $r \in RS$ do
 6:     if $\mu(c_{min}), \mu(s_{min}), \mu(b_{min}), \mu(dl), \mu(nl)$ fit the
    membership levels of $r$ then
 7:         Determine the output linguistic level $L$ from $r$
 8:         $\vartheta \leftarrow \min\ (\mu(c_{min}),\mu(s_{min}),\mu(b_{min}),\mu(dl),\mu(nl))$
 9:         $O \leftarrow O \cup \{(L, \vartheta)\}$
10:     end if
11: end for
12: Aggregate $O$ for all $L$ using the Maximum Aggregation
    Method
13: Defuzzify $O$ using Center of Gravity method to get the
    crisp value $D$
14: if $D > D_{th}$ then
15:     $Decision$ = Cloud
16: else
17:     $Decision$ = Fog
18: end if
19: return $Decision$
```

Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.

decision threshold D_{th} to decide on whether or not to offload the task to the cloud layer, in lines 14-19.

Algorithm 2 Fuzzy Logic-based Real Time Scheduling Algorithm (FLRTS)

```
    Input: $t$
    Output: $z(t)$
    Predefined: $V$, $nl$, $Q$, $FQ$
 1: for each $t$ in $Q$ do
 2:     if resource requested not provided by fog then
 3:         Forward the task to the cloud
 4:     else
 5:         Determine $c_{min}(t)$, $s_{min}(t)$, $b_{min}(t)$ by Eq. (4)
 6:         Normalize $c_{min}(t)$, $s_{min}(t)$, $b_{min}(t)$, $dl(t)$, $nl(t)$
 7:         $Decision \leftarrow$ FLDA ($c_{min}(t)$, $s_{min}(t)$, $b_{min}(t)$, $dl(t)$, $nl(t)$)    ▷ Alg. 1
 8:         if $Decision$ = Cloud then
 9:             Forward the task to the cloud
10:         else
11:             Insert the task in $FQ$
12:             Order the tasks in $FQ$ in ascending order of $dl(t)$
13:             for each task $t_i$ in $FQ$ do
14:                 Initialize $V^{'} \leftarrow \emptyset$
15:                 for each virtual machine $v_j$ in $V$ do
16:                     if ELIG$(t_i, v_j) = True$ then    ▷ Alg. 3
17:                         $V^{'} \leftarrow V^{'} \cup \{v_j\}$
18:                     end if
19:                 end for
20:                 for each virtual machine $v_j$ in $V^{'}$ do
21:                     Calculate resource utilization $ru(v_j)$
22:                     Obtain least loaded virtual machine $v^{'}$
23:                 end for
24:                 Forward the task $t_i$ to $v^{'}$
25:             end for
26:         end if
27:     end if
28:     Obtain the result of $t_i$ (from Cloud or Fog)
29:     Calculate the total processing time $T_{proc}(t_i)$
30:     if $T_{proc}(t_i) \leq dl(t_i)$ then
31:         $z(t) \leftarrow 1$
32:     else
33:         $z(t) \leftarrow 0$
34:     end if
35: end for
36: return z(t)
```

B. Fuzzy Logic-based Real-time Task Scheduling Algorithm

The tasks arriving at the fog broker are stored in a queue Q which is processed by the proposed *Fuzzy Logic-based Real-time Task Scheduling Algorithm* (Algorithm 2). If the resources required for the task are not provided by the fog layer, then the task is forwarded to the cloud (lines 1-3). Otherwise, the minimum values of the resource utilization requirements of the task are calculated and normalized using

Algorithm 3 Check Eligibility of VM for the Task (ELIG)

```
1: function ELIG(task $t_i$, virtual machine $v_j$)
2:     if $ct(t_i) \leq cc(v_j)$ and $st(t_i) \leq sc(v_j)$ and $bt(t_i) \leq bc(v_j)$ then
3:         return $True$
4:     end if
5: return $False$
6: end function
```

the min-max normalization (lines 5-6). The *Fuzzy Logic-based Decision Algorithm* (Algorithm 1) is used to decide whether to forward the task to the cloud or process on the fog based on the resource utilization of the task, the task deadline, and the network latency (line 7, Algorithm 2). If the task is allocated to the fog layer, then it is added to a fog queue FQ. All the tasks in the fog queue are ordered in the ascending order of their deadlines (lines 11-12). This helps process the urgent tasks with hard deadline first, thus ensuring that more tasks are completed within the deadline. Then, a set of eligible virtual machines $V^{'}$ is created for each task by checking the eligibility of each virtual machine. Eligible virtual machines that satisfy the resource demands of the task are determined using Algorithm 3 (lines 13-19, Algorithm 2). The least loaded virtual machine $v^{'}$ among the eligible virtual machines is selected and the task is assigned to it (lines 20-25). The total resource utilization of a virtual machine can be determined from the individual resource utilization as follows.

$$ru(v_j) = \frac{(cu(v_j) * cu(v_j) * cu(v_j))}{3} \tag{7}$$

$$\text{where } cu(v_j) = \frac{\sum_{i=1}^{N} ct(t_i)}{cc(v_j)}$$

$$su(v_j) = \frac{\sum_{i=1}^{N} st(t_i)}{sc(v_j)} \text{ and } bu(v_j) = \frac{\sum_{i=1}^{N} bt(t_i)}{bc(v_j)}$$

The least loaded virtual machine is one with the minimum total resource utilization by all tasks assigned to it, i.e., $v^{'} = \min_{v_j \in V^{'}} ru(v_j)$. When the task reaches the virtual machine chosen for processing, it waits in a queue based on its deadline (non-decreasing order), if the virtual machine is busy with another task. After the task is executed in the cloud or fog layer, the results are sent to the fog broker, which calculates the total processing time for the task. If the total processing time is less than the deadline, the task is considered as successfully executed, i.e., $z(t) = 1$. Otherwise, it is considered as a failure or rejection, i.e., $z(t) = 0$ (lines 28-34).

V. Performance Evaluation by Simulation

This section reports the performance evaluation of the proposed algorithm through simulation study.

A. Simulation Environment

We used the ***iFogSim*** simulator [26][27] to implement and evaluate our proposed algorithm. The ***iFogSim*** is a java-based

Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.

simulation toolkit developed on top of the the *CloudSim* simulator framework. Further, we integrate our fuzzy logic scheme with *iFogSim* using the *jFuzzylogic* open source java library [28]. All the results have been averaged over 50 independent simulation runs. Simulation experiments were conducted on a PC with Intel(R) Core(TM) i-7 2.40 GHz and 8 GB of RAM with OS Windows 7.

The infrastructure configurations for the simulations are detailed in Table III. The tasks are configured randomly as detailed in Table IV. The network latency between the fog and cloud layers is chosen randomly between $300ms$ and $800ms$. The expected execution time of the task is computed assuming the CPU rate of the virtual machines as 1500 MIPS (million instructions per second). The predefined decision threshold D_{th} is set as 0.5.

TABLE III: Resource Configurations of Fog-Cloud Infrastructure

Processing node type	**CPU Rate (MIPS)**	**Memory (MB)**	**Storage (GB)**	**Bandwidth (Mbps)**
Cloud	44800	40000	1000000	10000
Fog broker	22800	10000	10000	10000
Computational VM	1500	1500	1000	1000
Storage VM	1000	1000	1500	1000
Bandwidth VM	1000	1000	1000	1500
Standard VM	1500	1500	1500	1500

To evaluate the performance of the proposed Fuzzy Logic-based Real-time Task Scheduling (FLRTS) algorithm, we carried out simulation experiments by varying the number of tasks and virtual machines in the system. The results are compared with the standard *First In First Out* (FIFO) and *Short Job First* (SJF) scheduling algorithms, and the existing Real-Time Task Processing (RTP) algorithm proposed in [8]. We conducted two sets of experiments to analyze (1) the effect of the number of tasks using 12 virtual machines (Fig 3), and (2) the effect of the number of virtual machines with 60 tasks (Fig 4) and 300 tasks (Fig 5).

TABLE IV: Features of Tasks

Feature	**Min Value**	**Max Value**
Number of Instructions	10 MI	1700 MI
Deadline	1000 ms	2200 ms
Required Storage	10GB	1700 GB
Required bandwidth	10 Mbps	1700 Mbps

B. Performance Metrics

The metrics used for comparing the performance of the algorithms are Makespan (in seconds), Delay rate, Success ratio, Average Turnaround Time (in seconds) and Average Processing Time (in seconds). They are defined below.

Makespan is the time required to complete all the tasks in the system [15] [29]. It is computed as $Makespan = \max_{1\leq j\leq K}(\sum_{i=1}^{N}\frac{in(t_i)}{cc(v_j)})$ where N is the total number of tasks and K is the total number of virtual machines in the system. The terms $in(t_i)$ and $cc(v_j)$ represent the number of instructions in task t_i and the CPU rate of the virtual machine v_j respectively, as described in Section III-A. *Success Ratio* is the fraction of tasks executed within their deadlines [9]. *Average Turnaround Time* is the average time a task spends at the virtual machine, i.e., from the virtual machine receiving the task to sending back the result to the fog broker. *Delay Rate* is the average of the ratio of the delay in the execution of the task to the expected execution time of the task [8]. The expected execution time is the minimum time required to process the task in the fog layer, by considering the maximum available CPU rate. The delay is the additional time taken for the execution of the task than the expected time. It can be calculated as the difference between the turnaround time and the expected execution time. *Average Processing Time* is the average time taken from submission of the task by an IoT device to receiving back the processed result. Various components of the processing time are described in Eq. (1).

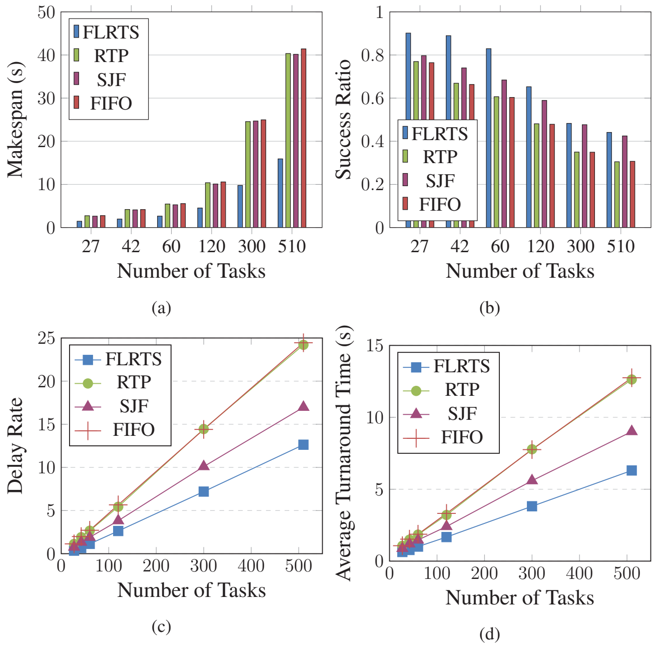


Fig. 3: Comparison of the algorithms based on the number of tasks (with 12 virtual machines)

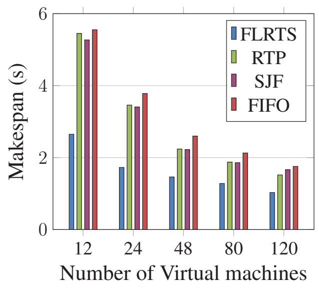
C. Experimental Results and Performance Comparison

Fig 3 compares the proposed FLRTS algorithm with three other algorithms using the metrics makespan, delay rate, success ratio, and average turnaround time with respect to number of tasks using 12 virtual machines. As the number of tasks increases, the time taken to process them also increases. This causes an increase in the delay, which in turn reduces the number of tasks executed successfully within the deadlines. Therefore, the success ratio decreases while the makespan, delay rate and average turnaround time increase with the number of tasks. The SJF and FIFO algorithms forward the task to the cloud only when the required resources are not available in the fog layer. The RTP algorithm also forwards the task to the cloud when the length (expected execution time) of the task is greater than a predefined threshold. However, the FLRTS algorithm considers the task deadline and transmission delay along with the availability of required resources at the fog layer. Further, the tasks may be forwarded to the cloud

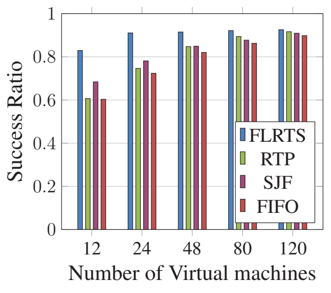
Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.

when the tasks have soft deadlines, or the latency between fog and cloud layer is minimal. Therefore, the number of tasks executed at the fog layer is less in FLRTS algorithm when compared with the other three algorithms.

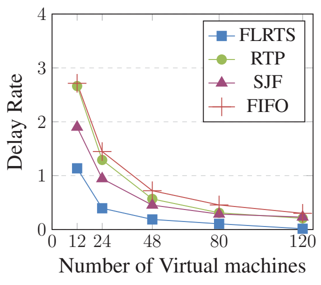
Additionally, the FLRTS algorithm finds the least loaded virtual machine in the fog layer for task execution. Consequently, the time taken to process the tasks and the delay is also smaller. Thus, the makespan, delay rate and average turnaround time for the FLRTS algorithm are lower than the other algorithms as shown in Fig. 3a, Fig. 3c and Fig. 3d. Moreover, the FLRTS algorithm considers the task deadline while scheduling the tasks, making the urgent tasks with hard deadline to be processed first. This enables FLRTS algorithm to give higher success ratio when compared with other algorithms as observed in Fig. 3b.



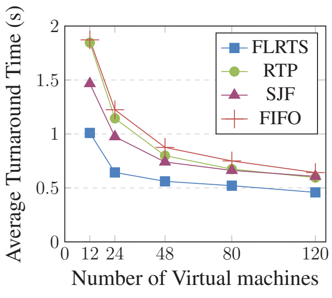
(a)



(b)



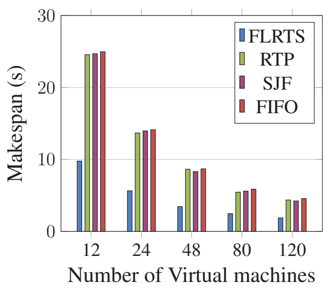
(c)



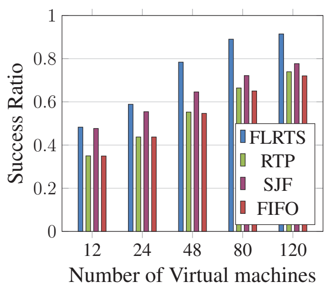
(d)

Fig. 4: Comparison of the algorithms based on the number of virtual machines (with 60 tasks)

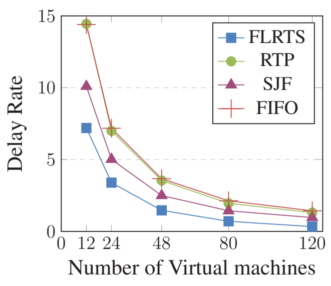
The resources available in the fog layer is proportional to the number of virtual machines. Thus, an increase in the number of virtual machines indicates the availability of more resources for processing the tasks in the fog layer. This causes the tasks to be spread out among the virtual machines. Therefore, we observe a decrease in the makespan, the delay rate and the average turn around time of the tasks as exhibited in Fig. 4a, Fig. 4c, Fig. 4d and Fig. 5a, Fig. 5c, Fig. 5d. Additionally, we observe a rise in the success ratio (see Fig. 4b and Fig. 5b). In the fog layer, existing SJF, RTP and FIFO algorithms can execute tasks based on the availability of required resources (for the tasks originated from IoT devices). However, the FLRTS algorithm considers the task deadline as another important parameter along with the resource availability. In case a task has a soft deadline and the communication latency of the fog-cloud link is very low, the FLRTS algorithm may forward the task to the cloud. Moreover, FLRTS gives better performance



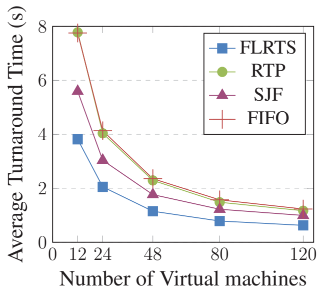
(a)



(b)



(c)



(d)

Fig. 5: Comparison of the algorithms based on the number of virtual machines (with 300 tasks)

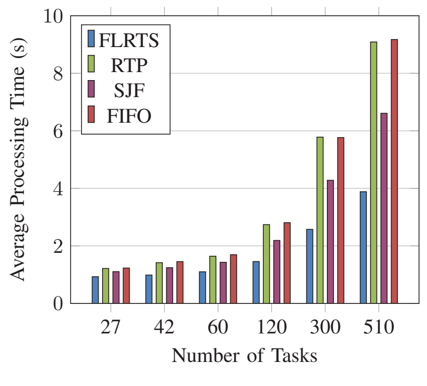
(with a given number of virtual machines) due to the fewer number of tasks being executed at the fog layer than in SJF, RTP and FIFO algorithms. This is due to the fuzzy decision for selection of some fraction of tasks to be forwarded to the cloud layer for better efficiency.

If there are fewer virtual machines, more tasks are allocated to each virtual machine, thus increasing the waiting time. However, increasing the number of virtual machines (although it depends on the available resources at the fog layer) minimizes the processing delay of the tasks by assigning them to different virtual machines. It can be observed from Fig. 4 and Fig. 5 that with the presence of a limited number of virtual machines, the proposed FLRTS algorithm outperforms the other three algorithms. As the the number of virtual machines increases, the overall performance of the system further improves as the task requirements are fulfilled with the availability of more resources at the fog layer. Thus, the overall performance of the fog-cloud system for IoT applications is improved using the proposed fuzzy logic based scheduling algorithm.

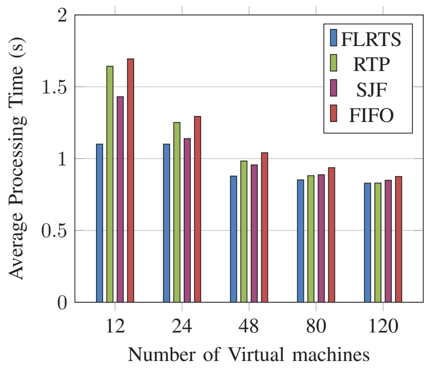
Fig. 6 compares the average processing time of the tasks by the proposed algorithm vis-a-vis existing SJF, RTP and FIFO algorithms. It is evident that FLRTS requires less processing time in comparison with other algorithms. The number of the tasks assigned to the fog layer by the FLRTS algorithm is fewer than those by other algorithms, thus reducing the turnaround time at the fog layer. As the number of tasks increases, the number of tasks assigned to the fog layer also increases. This in turn affects the waiting time of the resource availability at the fog layer.

Furthermore, for some tasks, the transmission time of the task to the cloud is smaller than the waiting time for the resources at the fog layer. In such cases, the FLRTS algorithm assigns the task to the cloud unlike other algorithms. There-

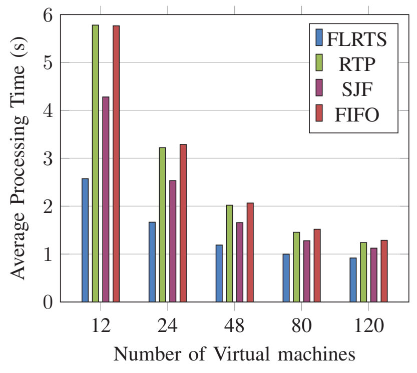
Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.



(a) with 12 virtual machines



(b) with 60 tasks



(c) with 300 tasks

Fig. 6: Comparison of the algorithms based on the processing time

fore, the processing time for the tasks is smaller than that of the other algorithms, as shown in Fig. 6a. By increasing the number of virtual machines in the fog layer, the resources available are also increased, hence reducing the waiting time for a task assigned to the fog layer to obtain the required resources. This causes a decrease in the processing time of the task with an increase in the number of virtual machines in the fog layer as shown in Fig. 6b and Fig. 6c. Additionally, the proposed FLRTS algorithm assigns the tasks in the fog layer to the least loaded virtual machine, causing a further reduction in the waiting time of the tasks. Therefore, we observe that FLRTS has the least processing time and hence performs better than other algorithms.

VI. Conclusion

In this paper, we have presented a real-time task scheduling algorithm in a heterogeneous fog-cloud environment integrated with a fuzzy logic based decision technique to appropriately distribute the tasks between the fog and cloud layers. We have considered the resource demands and the time constraints of the tasks in addition to the resource capacity and availability of the fog nodes to assign the task to a virtual machine in the fog layer. Simulation experiments have been carried out to evaluate the performance of the proposed algorithm. The results have been compared with other existing algorithms. Comparing the performance with respect to such metrics as the makespan, delay rate and average turnaround time, it has been observed that the proposed algorithm performs better than other algorithms. Moreover, the proposed algorithm assigns the tasks in the fog layer to the least loaded virtual machine and in-turn reduces the waiting time of the tasks.

In future, we plan to evaluate the proposed fuzzy logic based scheduling algorithm in a large-scale network environment and analyze its performance. Another direction of research is to integrate in the decision algorithm the mobility and privacy features of the tasks while allocating them between the fog and cloud layers. Yet another direction could be the investigation of channel bandwidth between the IoT devices and fog servers for variable data chunks (tasks) with a goal to determine the variation of the overall transmission time.

Acknowledgements: The authors are grateful to the reviewers for constructive comments that helped improve the technical and presentation quality of the manuscript significantly. The work of S. K. Das is partially supported by the US National Science Foundation grants CNS-2008878 and SaTC-2030624.

References

[1] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future generation computer systems*, vol. 56, pp. 684–700, 2016.

[2] M. De Donno, K. Tange, and N. Dragoni, "Foundations and evolution of modern computing paradigms: Cloud, iot, edge, and fog," *IEEE Access*, vol. 7, pp. 150 936–150 948, 2019.

[3] P. Bellavista, J. Berrocal, A. Corradi, S. K. Das, L. Foschini, and A. Zanni, "A survey on fog computing for the internet of things," *Pervasive and mobile computing*, vol. 52, pp. 71–99, 2019.

[4] S. Saraswat, H. P. Gupta, T. Dutta, and S. K. Das, "Energy efficient data forwarding scheme in fog-based ubiquitous system with deadline constraints," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 213–226, 2019.

[5] G. Zhang, F. Shen, Z. Liu, Y. Yang, K. Wang, and M.-T. Zhou, "Femto: Fair and energy-minimized task offloading for fog-enabled iot networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4388–4400, 2018.

[6] L. Yin, J. Luo, and H. Luo, "Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing," *IEEE Trans. on Industrial Informatics*, vol. 14, no. 10, pp. 4712–4721, 2018.

[7] L. Li, Q. Guan, L. Jin, and M. Guo, "Resource allocation and task offloading for heterogeneous real-time tasks with uncertain duration time in a fog queueing system," *IEEE Access*, vol. 7, pp. 9912–9925, 2019.

[8] S. Yin, J. Bao, J. Li, and J. Zhang, "Real-time task processing method based on edge computing for spinning cps," *Frontiers of Mechanical Engineering*, vol. 14, no. 3, pp. 320–331, 2019.

[9] N. Auluck, O. Rana, S. Nepal, A. Jones, and A. Singh, "Scheduling real time security aware tasks in fog networks," *IEEE Transactions on Services Computing*, vol. 14, no. 8, 2019.

Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.

[10] Z. Chen, Y. Zhu, Y. Di, and S. Feng, "A dynamic resource scheduling method based on fuzzy control theory in cloud environment," *Journal of Control Science and Engineering*, vol. 2015, no. 1, pp. 1–10, 2015.

[11] R. R. Rout, S. Vemireddy, S. K. Raul, and D. Somayajulu, "Fuzzy logic-based emergency vehicle routing: An iot system development for smart city applications," *Computers & Electrical Engineering*, vol. 88, p. 106839, 2020.

[12] M. A. Benblidia, B. Brik, L. Merghem-Boulahia, and M. Esseghir, "Ranking fog nodes for tasks scheduling in fog-cloud environments: A fuzzy logic approach," in *15th International Wireless Communications & Mobile Computing Conference (IWCMC)*. IEEE, 2019, pp. 1451–1457.

[13] M. D. Hossain, T. Sultana, V. Nguyen, T. D. Nguyen, L. N. Huynh, E.-N. Huh *et al.*, "Fuzzy based collaborative task offloading scheme in the densely deployed small-cell networks with multi-access edge computing," *Applied Sciences*, vol. 10, no. 9, p. 3115, 2020.

[14] H. Rafique, M. A. Shah, S. U. Islam, T. Maqsood, S. Khan, and C. Maple, "A novel bio-inspired hybrid algorithm (nbiha) for efficient resource management in fog computing," *IEEE Access*, vol. 7, pp. 115 760–115 773, 2019.

[15] B. M. Nguyen, H. Thi Thanh Binh, B. Do Son *et al.*, "Evolutionary algorithms to optimize task scheduling problem for the iot based bag-of-tasks application in cloud–fog computing environment," *Applied Sciences*, vol. 9, no. 9, p. 1730, 2019.

[16] T. Choudhari, M. Moh, and T.-S. Moh, "Prioritized task scheduling in fog computing," in *Proceedings of the ACMSE 2018 Conference*. ACM, 2018, p. 22.

[17] X. Li, J. Wan, H.-N. Dai, M. Imran, M. Xia, and A. Celesti, "A hybrid computing solution and resource scheduling strategy for edge computing in smart manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 15, pp. 4225–4234, 2019.

[18] T. S. Nikoui, A. Balador, A. M. Rahmani, and Z. Bakhshi, "Cost-aware task scheduling in fog-cloud environment," in *2020 CSI/CPSSI International Symposium on Real-Time and Embedded Systems and Technologies (RTEST)*. IEEE, 2020, pp. 1–8.

[19] M. Taneja and A. Davy, "Resource aware placement of iot application modules in fog-cloud computing paradigm," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2017, pp. 1222–1228.

[20] A. Mebrek, L. Merghem-Boulahia, and M. Esseghir, "Efficient green solution for a balanced energy consumption and delay in the iot-fog-cloud computing," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2017, pp. 1–4.

[21] S. Guo, L. Peters, and H. Surmann, "Design and application of an analog fuzzy logic controller," *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 4, pp. 429–438, 1996.

[22] I. Sakti, "Methodology of fuzzy logic with mamdani fuzzy models applied to the microcontroller," in *2014 The 1st International Conference on Information Technology, Computer, and Electrical Engineering*. IEEE, 2014, pp. 93–98.

[23] N. Dhanya, G. Kousalya, P. Balarksihnan, and P. Raj, "Fuzzy-logic-based decision engine for offloading iot application using fog computing," in *Handbook of Research on Cloud and Fog Computing Infrastructures for Data Science*. IGI Global, 2018, pp. 175–194.

[24] S. K. Mothku and R. R. Rout, "Adaptive fuzzy-based energy and delay-aware routing protocol for a heterogeneous sensor network," *Journal of Computer Networks and Communications*, vol. 2019, 2019.

[25] M. Masoum and E. Fuchs, *Optimal Placement and Sizing of Shunt Capacitor Banks in the Presence of Harmonics*. Elsevier, 12 2015, pp. 887–959.

[26] R. Mahmud and R. Buyya, "Modelling and simulation of fog and edge computing environments using ifogsim toolkit," *Fog and edge computing: Principles and paradigms*, pp. 1–35, 2019.

[27] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275–1296, 2017.

[28] P. Cingolani and J. Alcalá-Fdez, "jfuzzylogic: a java library to design fuzzy logic controllers according to the standard for fuzzy control programming," *International Journal of Computational Intelligence Systems*, vol. 6, no. sup1, pp. 61–75, 2013.

[29] Q. Liu, Y. Wei, S. Leng, and Y. Chen, "Task scheduling in fog enabled internet of things for smart cities," in *2017 IEEE 17th International Conference on Communication Technology (ICCT)*. IEEE, 2017, pp. 975–980.

Authorized licensed use limited to: C. V. Raman Global University - Bhubaneswar. Downloaded on August 09,2024 at 09:08:33 UTC from IEEE Xplore. Restrictions apply.