
Downloaded from: https://e-space.mmu.ac.uk/626654/
Version: Accepted Version
Publisher: Institute of Electrical and Electronics Engineers (IEEE)
DOI: https://doi.org/10.1109/jiot.2020.3022322

Please cite the published version
RTOS: Response-aware Traffic Offloading Scheme using Regression Machine Learning for User-Centric Large-Scale Internet of Things

Gunasekaran Manogaran, Gautam Srivastava (Senior Member, IEEE), Bala Anand Muthu, S. Baskar, P. Mohamed Shakeel, Ching-Hsien Hsu, Ali Kashif Bashir (Senior Member, IEEE), Priyan M. Kumar

Abstract—Resource allocation and management in an Internet of Things (IoT) paradigm requires precise request and response processing irrespective of its scalability support. Unpredictable traffic pattern and user density demands reliable offloading for handling user request traffic and service response. Considering the need for large-scale IoT in account of its interoperability and heterogeneous support, this manuscript introduces response-aware traffic offloading scheme (RTOS) for delay-sensitive user requests. This offloading scheme is supported by multivariate spline regression machine learning model for classifying traffic for reducing the failure rate. The splines are adaptive based on the classified traffic for performing independent and shared offloading. The computation process for determining the offloading model is inherited from the cyber physical system (CPS) coupled with the IoT-Cloud architecture. The information from the knowledge base and event logs are exploited for decision-making in employing the offloading method for the classified traffic. The simulation analysis of this scheme shows that it is effective in improving the request processing ratio and reducing processing, response time, and delay. The simulation is performed for the varying user density and traffic flows.

Index Terms—Basis Function, Cyber Physical System, IoT, Multivariate Spline Regression, Traffic Offloading

I. INTRODUCTION

Internet of Things (IoT) has emerged as a smart object enabling technology for meeting the demands of increasing user requests. The exceptional features of the IoT paradigm endorse platform flexibility, service scalability and ubiquitous access to resources due to incorporation of diverse information communication technologies (ICT) [1, 2]. With the growth of human population and service requirements, the need for extending IoT classes from supporting a small scale to large-resource environment becomes necessary. Though the scalability and flexibility of the decentralized communication and computation paradigm is improved, service response and quality of service compliance needs to be retained in a more reliable manner [3, 4]. Smart objects and ICT jointly construct the platform for IoT users to improve the on-demand communication, information sharing and process computations. Besides, IoT paradigm provides global access to resources through interoperability with cloud, mobile edge computing (MEC), fog nodes, smart sensors, etc. [2, 4]. Envisioning the information and user requests in digital form enables machine-to-machine, device-to-device, human-computer interaction, etc., depending upon the application area ranging from residential to commercial and healthcare industries [5].

The density of request and traffic in a large-scale IoT is unpredictable due to which processing and resource allocation failures are common [6]. The conventional computation process and resource allocation systems inherit the advantages of the attached infrastructure support for handling the varying traffic patterns [7]. For handling varying traffic patterns, MEC, cloud, and fog based architectures are introduced for IoT paradigm. The computations are performed in a distributed and shared manner between the associated systems [5, 8]. Offloading is a prominent process for suppressing the traffic and congestion in the ubiquitous environment. Storage and computation, architecture based offloading schemes have been designed for smoothing IoT traffic in a view to improve the performance and reliability of resource allocation and sharing systems [6, 9]. Task or workflow offloading follows the conventional scheduling process for handling overloaded requests and service responses. Delay-sensitive application requirements in the large-scale environment are to be responded without time-lapse to retain the service quality of the users [5, 8, 10].

Cyber Physical System (CPS) has evolved as an integral support for improving the service quality of autonomous and distributed communication environments. Diverse computing, communication, and control systems are unified under CPS that

Gunasekaran Manogaran is with University of California, Davis, USA (e-mail: gmanogaran@ieee.org).
Gautam Srivastava is with the Dept of Math and CSC, Brandon University, Canada (e-mail: srgvastavag@brandonu.ca).
Bala Muthu is with VRS College of Engineering and Technology, India (e-mail: balaanand@vrscet.in).
S. Baskar is with Karpagam Academy of Higher Education, India (e-mail: basker.s@kahedu.edu.in).
P. Mohamed Shakeel is with the University of Technical Malaysia Melaka (e-mail: shakeel@utem.edu.my).
Ching-Hsien Hsu is with Asia University, Taiwan (e-mail: robertchh@asia.edu.tw).
Priyan M K is with Middlesex University, UK (e-mail: P.Malavizhikumar@ieee.org).
Ali Bashir is with the Department of Computing and Mathematics, Manchester Metropolitan University, UK (e-mail: sh_alikashx1@ieee.org).
is coupled to the IoT paradigm [11, 12]. Integration of such control system aids multi-faced benefits for IoT by aiding dynamic and reconfigurable service support and reliable resource oriented functions. The benefits of CPS are directly inherited in the IoT environment for improving the service response, delay-sensitive processing, resource allocation, and heterogeneity and scalable service support [7, 13]. The benefit of the integration of CPS into IoT-cloud is the independent and concurrent processing ability and distributed access. CPS is employed for resource allocation, traffic smoothing, data analytics and representation, visualizing events, data management, service compliance, etc. [12, 14].

II. RELATED WORKS

Kim et al. [15] introduced adaptive job allocation scheduler (AJAS) for redistributing overloaded IoT tasks. Tasks are distributed on the basis of user behavior, resource performance and energy exhaustion rate. This kind of allocation improves the availability of devices for processing the overloaded tasks.

Energy-cost dependent task offloading in heterogeneous IoT is introduced by Jaddoa et al. [16]. Based on the cloud/edge node state and available network connections, dynamic decisions are made for offloading the arriving task. This decision-making process is reliable in improving the energy-efficiency and ratio of processed tasks in the IoT-cloud-edge architecture.

Fair and energy-minimized task offloading (FEMTO) is designed by Zhang et al. [17] for improving the performance of fog-assisted IoT networks. Offloading of tasks is performed by considering energy utilization, average energy exploitation in the previous states, and the priority of the fog nodes. This offloading method improves the fairness index along with probability of overloading under controlled energy consumption.

Sun et al. [18] projected energy and time efficient task offloading and resource allocation algorithm for IoT-fog-cloud architecture. This algorithm is designed to reduce the energy and cost of through local computation and transmission power allocation. This algorithm achieves less energy utilization and computation time with better energy efficiency cost.

Misra and Saha [19] proposed dynamic task offloading for software-defined network assisted fog paradigm supporting IoT. The offloading method follows reliable decision-making, node discovery and route selection for aiding better performance. Offloading problem is addressed as an integer linear program in order to reduce delay and energy usage.

In order to handle computation demanding IoT tasks, Stavrinides and Karatza [20] designed hybrid flow-flow scheduling method. This method is effective for fog assisted IoT architectures, to reduce the computation cost and to smooth the offloading process. The proposed offloading method achieves better cost and resource utilization.

Kim et al. [21] introduced adaptive packet scheduling using Q-learning (APA-QL) for handling variable traffic flows in IoT environment. Q-learning helps to gain knowledge of the gateway queue size for scheduling the unpredictable traffic packet flow. Therefore, the processing time, computation process is reduced in this method. This method requires update of Q-table that increases the stagnancy in processing packets.

To handle deadline and cost dependent cloud workflows, Ma et al. [22] designed and IoT based effective task scheduling scheme. This task scheduling scheme identifies the performance of the virtual machine for satisfying the cost and delay constraints of the workflows. The allocation and scheduling process is facilitated through genetic process in order to improve the diversity of scheduling process.

Zhang et al. [23] employed cuckoo search algorithm for resource scheduling in IoT. The process of resource scheduling and allocation is differentiated from the premature convergence issue through Cauchy mutation. This helps to accelerate the number of local search, and the global solutions for scheduling. The heterogeneous resource scheduling algorithm improves resource utilization and relativity of user requests.

Boveiri et al. [24] employed min-max ant system (MMAS) for optimal task scheduling in cloud assisted IoT applications. This system employs traditional ant colony optimization for scheduling tasks over multi-processor systems. The design aim of this system is to identify the priority of the tasks for allocating appropriate systems for processing.

For edge assisted IoT services, Alameddine et al. [25] designed dynamic task offloading and scheduling (DTOS) method. This scheduling method is reliable in reducing the complexity in allocating resources through logic-based benders decomposition (LBBD). In this problem solving method, the primary and secondary computation models are useful in achieving optimal solutions for delay-sensitive applications.

Luo et al. [26] introduced adaptive task offloading (ATO) method for improving the performance of mobile edge computing paradigm in industrial applications. Aided by CPS, this offloading method is based on auction to meet the adaptability and security requirements of the edge server. The offloading method is found to achieve high resource utilization and controlled delay.

Energy cost and congestion issues in CPS based edge computing are jointly optimized by Yang et al. [27]. Promoted-by-probability (PBP) scheme is used in this optimization model for overcoming the fore-mentioned issues by classifying packets based on priority. Krill Herd Meta heuristic optimization algorithm used in the PBP helps to reduce the complexity of edge computing systems.

Yin et al. [28] designed a real-time task processing method using the collaboration of edge computing and CPS. High speed processing and response are the design objective of this collaborative architecture where task and control flow between the communication and CPS layers are handled independently. Queue aware scheduling and task assignment of this method helps to reduce delay and to improve the resource utilization rate.

III. RESPONSE-AWARE TRAFFIC OFFLOADING SCHEME (RTOS)

The design goal of RTOS is to improve the reliability of request processing and thereby to improve the response service
quality in large-scale IoT. The computation model required to concise requests and response is distributed among the CPS-coupled IoT systems. CPS assists recommendation in handling responses for the overloaded and congested user requests. Divided by time and service based user demands, response is spontaneous satisfying the delay-less and backlogs-free compliance. The request traffic is in an independent and shared manner as distinguished by the capacity of the IoT computing system. The process of RTOS is portrayed in Figure 1.

![Figure 1 RTOS Process in IoT-Cloud-CPS Architecture](image)

In Figure 1, RTOS process is illustrated in an IoT-Cloud-CPS architecture. The process of Traffic offloading is independent between the end-user devices and cloud architecture. In case of a shared offloading, CPS offers traffic handling features. Therefore, the offloading is determined with the aid of two-assistive architectures of IoT. End-user devices/ applications interact with the cloud/ CPS through the communication infrastructure. This architecture is designed for large-scale IoT request-response management, where traffic due to request and resource (service) is common, unattended/ non-smoothed traffic results in failure of request processing, delayed response and less feasible quality of service. In the following section, the rate of traffic generated and the appropriate offloading process is discussed in detail.

### Traffic Generation Rate

The density \( \rho \) of users vary with different instance of time where in \( \{r_1, r_2, \ldots, r_p\} \) is the number of requests generated. The transmission time \( t_r \) varies in the processing time \( p_t \) of the IoT service provider. In a large-scale IoT, definite time slots for \( r_p \) transmission cannot be predicted or formulated as the status (completed/pending) of the previously generated requests relay on the processing capacity \( p_t \) of the \( m \) service providers. Therefore, some requests from the previous \( t_r \) are carried forwarded in the successive processing time intervals. The maximum rate of traffic (requests) generated in \( \sum t_r = (\rho \times r_p \times t_r) \) for which the rate of response needs to be allocated in precise time. The maximum time until a user can wait is the validity time \( t_v \) of the request. If the response time \( t_r \) exceeds \( t_v \), then it is said to be a failure. Therefore, allocating appropriate resources for the received request traffic with excess wait time or failure defaces the quality of service. The traffic flow of the requests in a time \( t_r \) is computed by equation

\[
\emptyset_r = \begin{cases} \frac{\rho \times r_p \times t_r}{r_p - \frac{\rho}{r_p} \times t_v} \\ \text{such that} \quad (t_r + t_v) \leq t_v \end{cases}
\]  

(1a)

The traffic of the requests \( \emptyset_r \) is estimated for a single flow observed in \( t_r \). Determining the traffic flow with respect to \( \left( r_p - \frac{\rho}{r_p} \right) \) (the variation with respect to \( \rho \) and \( r_p \) is validated as a single IoT user/application is capable of generating multiple requests) helps determine the offloading type. Now, the objective of the proposed scheme is defined by equation (1b) as

\[
\forall \emptyset_r \text{ and } p_c > 0 \in t_r, \quad \text{minimize } (t_r - t_v) \forall \text{ independent offloading} \\
\text{minimize } (t_r - t_c + t_w) \forall \text{ shared offloading} \\
\text{such that, } \left( r_p - \frac{\rho}{r_p} \right) > 0 \text{ in all } t_r
\]

(1b)

In equation (1b), the factor \( t_w \) denotes the wait time of \( r_p \) for a shared offloading process. The factor \( \left( r_p - \frac{\rho}{r_p} \right) < 0 \) indicated that no requests are generated from \( \rho \) users and therefore the device traffic needs not be accounted. The objective in equation (1b) emphasizes the minimization of \( (t_r + t_v) \) and \( t_r + t_c + t_w \) to improve response rate. Therefore, the objective focuses to improve the response of all \( \left( r_p - \frac{\rho}{r_p} \right) < 0 \emptyset_r \) generated in \( t_r \). The \( t_r \) rate of \( \emptyset_r \) if balanced to be served within the next \( t_r \) sequence, and then response time is less. The decision making in balancing \( \emptyset_r \) and satisfying equation (1b) is aided by adaptive regression spline learning. The following subsections describe the role of spline learning for independent and shared offloading process.

#### Independent Offloading

The process of independent offloading is carried out between the cloud resources through local decision-making. The decision-making focuses to maximize the response rate and to retain \( (t_r + t_v) \leq t_v \). If the above conditions are satisfied, then the ratio of processed requests is enhanced. The available \( m \) service providers are to be scheduled to accept \( r_p \) in time \( t_r \). Irrespective of the condition of \( \left( r_p - \frac{\rho}{r_p} \right) > 0 \), the arrived requests are to be responded. The \( p_c \) of the available \( m \) plays a vital role in determining \( t_r \). However, there exists a tradeoff between \( p_c \) and \( r_p \) (allocated for \( m \)) due to the process start time \( (t_r) \) and \( t_r \). This tradeoff factor \( (y_r) \) is defined.
using equation (2) as
\[
Y_{r} = \arg\min \left[ \frac{(r_{p} - p_{t}) \times (t_{r} - t_{s})}{(t_{r} - t_{s}) - (t_{s} - p_{t})} \right] + 1 - \left( \frac{p_{t} - t_{s}}{r_{p} - p_{t}} \right) \times (t_{r} - t_{s}) - (t_{s} - p_{t}) \tag{2} \]

In equation (2), the tradeoff between \( p_{t} \) and \( r_{p} \) is estimated with respect to the \( t_{s} \) and \( p_{t} \). Hence, the decision for identifying \( r_{p} \) that exceeds \( t_{s} \) (in terms of time) is necessary. The conventional limit of requests for shared offloading lies between \( \left[ \frac{t_{r}}{p_{t}} \times m \right] \), \( r_{p} \) for \( \phi_{r} \) received in \( t_{s} \). The range of \( r_{p} \) is the worst-case in handling \( \phi_{r} \) as the failure increases. However, reducing \( \left( \frac{t_{r}}{p_{t}} \times m \right) \) and \( r_{p} \) increases the changes of reliable request processing. The sensitivity of retaining \( (t_{s} + t_{r}) \leq t_{v} \) is achieved by reducing \( Y_{r} \) and \( \left( \frac{t_{r}}{p_{t}} \times m \right) \). The case of reducing \( r_{p} \) is achieved through shared offloading as \( \left( \frac{t_{r}}{p_{t}} \times m \right) < r_{p} \). Instead, if \( r_{p} = \frac{t_{r}}{p_{t}} \times m \), then \( u \)-independent offloading is sufficient. The decision in determining the range of the \( r_{p} \), reducing \( Y_{r} \) is progressed using adaptive splines. The required output as per the spline model for deciding \( \left( \frac{t_{r}}{p_{t}} \times m \right) \)

\[ (i.e.) \left( Y_{r_{p} p_{t} t_{r}} \right) \] is given by equation (3a)
\[
Y_{r_{p} p_{t} t_{r}} = (1 - \Delta) + \sum_{i=1}^{p_{t}} \left( 1 - \Delta_{i} \right) B_{i}(\phi_{r}) \tag{3a} \]

where,
\[
\Delta = \left( \frac{r_{p} - p_{t}}{r_{p}} \right) \text{ and } \quad B_{i}(\phi_{r}) = \left( t_{r_{1}} \times r_{p_{1}} \right) + \frac{t_{r_{2}}}{2} \times r_{p_{2}} + \cdots + \frac{t_{r_{p_{t}}}}{p_{t}} \times r_{p_{t}} \tag{3b} \]

Here, \( B_{i}(\phi_{r}) \) is the basis function for the requests processed through \( p_{t} \). In equation (3a) and (3b) the differential unit \( \Delta \) is used for validating all the process of the independent offloading. For the estimation of \( \left( \frac{t_{r}}{p_{t}} \times m \right) \) and the achievement of \( Y_{r} \), the possibility of \( Y_{r} \) in \( \phi_{r} \) is to be estimated with the consideration of \( \Delta \) and \( p_{t} \) of the service provider. In this case, the tradeoff is computed as the different between time-lapse for the \( r_{p} \) received and service generated. This is represented as
\[
Y_{r} = \begin{cases} m \left( \frac{t_{r} - p_{t}}{r_{p}} \right), & \text{if } \Delta > 0 \\ 0, & \text{if } \Delta \leq 0 \tag{4} \end{cases} \]

Substituting for \( Y_{r} \) from equation (2) and solving for 0 the RHS of equation (4)
\[
\Delta \times \frac{\left( t_{r} - p_{t} \right) m}{r_{p}} = \left( \Delta - 1 \right) \frac{p_{t}}{r_{p}} m \tag{5a} \]

In equation (5a), the \( p_{t} \) is substituted for \( (t_{r} - t_{s}) \) time in \( m \) and considering \( t_{w} = 0 \). Similarly, \( p_{c} > \frac{t_{r}}{p_{t}} m \), it can be neglected such that
\[
\Delta \left( t_{r} - r_{p} m \right) = (1 - \Delta) \frac{t_{r}}{p_{t}} m \tag{5b} \]

\[
\Delta \left( t_{r} - r_{p} \right) = m \left( \frac{t_{r}}{p_{t}} + \frac{\Delta}{r_{p}} \right) \tag{5c} \]

In the final value of \( Y_{r} \) in \( \phi_{r} \) at a time span of \( (t_{r} - t_{s}) \). The value is the trade off in equation (2) is reduced to the one represented in equation (5c). If the \( t_{s} \) and \( p_{t} \) are assumed to be constant, then \( \Delta \) am. Now, the validation of range of \( \left( \frac{t_{r}}{p_{t}} \times m \right) \) in \( \phi_{r} \), for \( Y_{r_{p} p_{t} t_{r}} \) is performed. From equation (3b), the basis function is modeled with consideration of the limit as \( (1, \frac{t_{r}}{p_{t}} m) \) such that,
\[
B_{i}(\phi_{r}) = \sum_{i=1}^{p_{t}} m \left( 1 - \Delta_{i} \right) \frac{t_{r_{i}}}{p_{t}} \tag{6a} \]

And
\[
Y_{r_{p} p_{t} t_{r}} = \sum_{i=1}^{p_{t}} \left( 1 - \Delta_{i} \right) \frac{t_{r_{i}}}{p_{t}} \tag{6b} \]

As the initial \( \Delta = 0 \) (before specifying the range of \( r_{p} \)),
\[
Y_{r_{p} p_{t} t_{r}} = \sum_{i=1}^{p_{t}} \left( 1 - \Delta_{i} \right) \frac{t_{r_{i}}}{p_{t}} \tag{6c} \]

The output of equation (6c) denotes the limit of \( \left( \frac{t_{r}}{p_{t}} m \right) \) in \( \phi_{r} \) such that \( (1 - \Delta) \) is the range determining factor. The process of offloading \( \left( \frac{t_{r}}{p_{t}} m \right) \) range of \( r_{p} \) is represented in Figure 2(a) and (b)

\[ \text{Figure 2 (a) Offloading for the range } \left[ \frac{t_{r}}{p_{t}} \times m \right] \]

The process in Figure 2(a) is a linear case, where the output of the spline is a multivariate, as \( Y_{r_{p} p_{t} t_{r}} \). This multivariate needs to be approximated using \( \Delta \), in order to allocate maximum \( r_{p} \) to the available \( m \) in \( (t_{r} - t_{s}) \) time. Therefore, the offloading range of \( r_{p} \) is computed as
\[
\text{offloading Range} = \sum_{i=1}^{p_{t}} \left( \frac{r_{p} t_{r} m}{\phi_{r}} \right) \left( \frac{\rho_{r} \Delta}{(\rho_{r})^{i}} x \right) (1 - \Delta_{i}^{2}) \tag{7} \]

In equation (7), the offloading range for \( r_{p} \) is determined and therefore, the actual range is [offloading range (as in equation (7)), \( r_{p} \)]. As the independent offloading follows the range from first request to the range in equation (7), the \( r_{p} \) from \( \phi_{r} \) that belongs to [1, offloading range] is the requests processed by \( m \) in \( (t_{r} - t_{s}) \). \( t_{r} > t_{s} \). The above range is estimated for \( \phi_{r} \), in \( t_{r} \) provided \( Y_{r} \) is less, if the \( \left( \frac{t_{r}}{p_{t}} \times m \right) \) is reduced to offloading range. This process is represented in figure 2(b).
cross validation of multivariate spline factors (1 − Δ² and tₚ₊₁(as Δ=0, in the first tₚ) using the basis function. This offloading ensures (tₑ – tₛ) ≤ tᵥ or tₑ ≤ tᵥ for the new offloading range. Hence, the processing of [∅ᵣ, x(1 − Δ²) x 𝛾ᵣ] following equation (5c) is said to be high. The estimation of 𝛾ᵣ is provided (τₑ − τₛ ≤ τᵥ) is to be satisfied dialing which results in extended delay and response time. Therefore, the assignment (offloading) of the remaining 𝜕𝜕 follows the extended case of 𝜕𝜕 as given below:

Considering that m service providers have different 𝑡ᵣ, the estimated of 𝜕𝜕 (not 0) is:

\[
\begin{align*}
\gamma_r(1) &= m \times \frac{p_t}{p_c} + \sum_{i=1}^{m} \left( \frac{q_i}{\beta_i} + \Delta \right) \\
\gamma_r(2) &= m \times \frac{p_t}{p_c} + \sum_{i=2}^{m} \left( \frac{q_i}{\beta_i} + \Delta t \right) \\
\gamma_r(q) &= m \times \frac{p_t}{p_c} + \sum_{i=q}^{m} \left( \frac{q_i}{\beta_i} + \Delta q-1 \right)
\end{align*}
\]

The estimation of 𝜕𝜕 for the 𝑚 range of 𝑝𝑐 and 𝑝𝑡 to improve the offloading rate, where in Δq is estimated as (Δq − Δq) > 0, provided (𝑡ᵥ + 𝑝𝑡 > 𝑡ᵥ). Therefore, (Δq × 𝜕𝜕) is the rate of failed requests. The conventional offloading in presented in Figure 3, for linear 𝜕𝜕 management.

**Shared Offloading**

The other range of 𝜕𝜕 (i.e) [offloading range, 𝑝𝑡] is handled through this process. In this case, the objective is to reduce 𝜕𝜕, which means not all the 𝜕𝜕 in time 𝑡ₑ is offloaded. This helps to reduce the unnecessary wait time and response delay of 𝜕𝜕. The available m is not computationally sufficient (in time 𝑝𝑡) for handling the new range of offloaded request traffic. Therefore, there are two concurrent processes in this offloading namely optimal reduction of the offloading limit and satisfying (𝑡ₑ − 𝑡ₛ + 𝑡ᵥ) or (𝑡ₑ − 𝑡ᵥ) ≤ 𝑡ᵥ. This demands a multi level decision making that is facilitated by CPS knowledge base. The knowledge base and event log elements of CPS is exploited for handling the shared offloading of 𝜕𝜕. The previous state (independent offloading) 𝜕𝜕 and 𝜕𝜕 are stored in the knowledge base and the event log held’s the remaining 𝜕𝜕 (i.e) \(Y_{r_p, p_t, p_c} - \gamma_r\) request traffic to be handled. Until an offloading range variation is experienced, shared offloading is not facilitated. The first come first serve function of the service provider is modified in a dependent and discrete manner through the local decisions of CPS. The aim of such computation is to maximize the ratio of offloaded 𝜕𝜕 processing without increasing the 𝑝𝑡. The idle time of the m is identified using 𝑝𝑡 and 𝑡ₛ to assign the segregated 𝜕𝜕 from the independent offloading process. As identified in the previous case, \(Y_{r_p, p_t, p_c} - \gamma_r\) is the offloaded traffic where in the 𝜕𝜕 experiences 𝑡ᵥ until the appropriate m is assigned. The decision making is performed on the basis of differential adaptive spline factors \(p_c, p_t\) for satisfying the objective as in equation (1 b), the case of 𝑝ₜ being the same for all the available ‘m’ is assumed in this offloading process such that 𝑝ₜ > 0 (tₑ − tₛ). Similar to the previous case, the 𝜕𝜕 minimization is considered for the remaining 𝜕𝜕. The basis function for \(p_c, p_t\) variates is estimated as

\[
B_i(\delta) = \sum_{i=q}^{m} \frac{p_t}{p_c} \frac{\psi}{(1 - \frac{\psi}{\delta})}
\]

(8a)

For the above basis function, the constraint of \((p_t + t_v) \leq t_v\) is to be satisfied dialing which results in extended delay and response time. Therefore, the assignment (offloading) of the remaining 𝜕𝜕 follows the extended case of 𝜕𝜕 as given below:

\[
\gamma_r(1) = m \times \frac{p_t}{p_c} + \sum_{i=1}^{m} \left( \frac{q_i}{\beta_i} + \Delta \right) \\
\gamma_r(2) = m \times \frac{p_t}{p_c} + \sum_{i=2}^{m} \left( \frac{q_i}{\beta_i} + \Delta t \right) \\
\gamma_r(q) = m \times \frac{p_t}{p_c} + \sum_{i=q}^{m} \left( \frac{q_i}{\beta_i} + \Delta q-1 \right)
\]

(9)

The estimation of 𝜕𝜕 for the 𝑚 range of 𝑝𝑐 and 𝑝𝑡 to improve the offloading rate, where in Δq is estimated as (Δq − Δq) > 0, provided (𝑡ᵥ + 𝑝𝑡 > 𝑡ᵥ). Therefore, (Δq × 𝜕𝜕) is the rate of failed requests. The conventional offloading in presented in Figure 3, for linear 𝜕𝜕 management.
the wait time of $\emptyset_r$ from $q$ to $\partial$ be $[\frac{Y_r}{\partial_r}(1 - \Delta^2) \times t_s - t_r]$. The traditional wait time of $(p_t - t_s)$ is reduced due to concurrent and shared offloading of $\partial$ as well under controlled wait time.

IV. PERFORMANCE ANALYSIS

The performance of the proposed RTOS is validated using simulations carried out in Contiki Cooja simulator. In coherence to the IoT-cloud-CPS architecture, a large-scale IoT environment comprising of 800 users is used in the simulation. The cloud is modeled with 8 servers each of 2 TB storage and 2×4GB physical memory. In this environment, the communication layer consists of 32 access points for connecting cloud, CPS and IoT user devices. In Table 1 (a) and 1 (b), the physical configuration of devices, cloud and CPS and simulation parameters and its values are discussed.

<table>
<thead>
<tr>
<th>Table 1 (a) Physical Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration</strong></td>
</tr>
<tr>
<td>Number</td>
</tr>
<tr>
<td>Physical Memory (Gb)</td>
</tr>
<tr>
<td>Storage Size (Gb)</td>
</tr>
<tr>
<td>CPU (GHz)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 1 (b) Simulation Parameters and Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Simulation Parameters</strong></td>
</tr>
<tr>
<td>Requests/ s</td>
</tr>
<tr>
<td>Request Validity Time (ms)</td>
</tr>
<tr>
<td>Processing Capacity of m (requests/s)</td>
</tr>
<tr>
<td>Computation Memory of CPS (Mb)</td>
</tr>
<tr>
<td>Event Log Memory (Mb)</td>
</tr>
<tr>
<td>Bandwidth (Mb/s)</td>
</tr>
</tbody>
</table>

The above configurations are used for performing both independent and shared offloading process for the varying traffic (request) flows. The metrics that are computed in this experimental setup are: Processing time, processing ratio, average delay, and response time. The metrics are estimated for varying traffic flows, user density. For a consistency verification, the proposed RTOS is compared with the existing DTOS-LBBD [25], APS-QL[21], and ATO [26].

4.1 Analysis of Processing Time

In Figure 4 (a)-4 (d), the comparative analysis of processing time with respect to varying traffic flows and users is presented. The joint $\emptyset_r$ processing time as in independent and shared offloading is estimated in the above presentation. The time required for processing $r_p$ in $\emptyset_r$ in the independent offloading is $(t_s - t_r)$ and in the shared offloading case, it is $(t_w + p_t)$ as the offloading is linear. This is replaced with the new multivariate factors $(p_c, p_t)$. The basis function confines the processing time to be not extended beyond $t_v$ such that $(t_s - t_r)$ and $(t_w + p_t)$ in independent and shared offloading satisfies the condition of $\leq t_v$. Therefore, the processing time of shared offloading of $\partial$ is congruent with the $Y_{\emptyset_r}^p$ in $\emptyset_r$.

Thus, the classified and approximated range of $\emptyset_r$ in both the offloading achieves less processing time. The processing time is balanced by altering the spline multivariate depending on $\partial$ and $p_c$ for reducing additional time for $Y_\emptyset$. 

![Figure 4 Processing Time](image-url)
4.2 Analysis of Processing %

There are two factors that augment the ratio of processing requests namely the spline multivariate determination and offloading based on CPS knowledge base. In the former factor, the adaptive feature of the regression model is exploited by varying the multivariate \( y_{r}, \frac{t_r}{p_t} \) to \( (p_c, p_t) \) depending on the rate of \( \emptyset_r \) processed. The classification and detection of \( \emptyset_r \) range through \( B_i(r_p) \) helps to reduce the failure in processing \( \emptyset_r \). The classification helps to maximize the chances of processing \( r_0 \) in \( \emptyset_r \) by defining the ranges. Therefore, it is common that processing ratio is better in the first offloading case. In Table 2, the comparative analysis of processing ratio with respect to the wait time (8ms to 16ms) is presented.

Table 2 Processing Ratio with respect to Varying Wait Time

<table>
<thead>
<tr>
<th>Wait Time (ms)</th>
<th>DTOS-LBBD</th>
<th>APS-QL</th>
<th>ATO</th>
<th>RTOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>47.31</td>
<td>61.25</td>
<td>96.15</td>
<td>83.86</td>
</tr>
<tr>
<td>10</td>
<td>47.62</td>
<td>61.46</td>
<td>69.31</td>
<td>83.98</td>
</tr>
<tr>
<td>12</td>
<td>47.89</td>
<td>61.76</td>
<td>69.52</td>
<td>84.02</td>
</tr>
<tr>
<td>14</td>
<td>48.03</td>
<td>62.04</td>
<td>69.89</td>
<td>84.16</td>
</tr>
<tr>
<td>16</td>
<td>48.1</td>
<td>62.82</td>
<td>71.5</td>
<td>86.48</td>
</tr>
</tbody>
</table>

In case of shared offloading the objective in equation (1b) is satisfied by altering the first come first serve \( r_0 \) processing to knowledge base and event log based recommendations. This recommendation considers \( p_c \) and \( (t_t + p_w) < t_r \) constraint to assign all the \( \emptyset_r \) from \( q \) to \( \emptyset \) to the available \( m \). Therefore, the rate of processing \( \emptyset_r \) is high in the shared offloading process [Refer Figure 5 (a) - 5 (d)].

4.3 Analysis of Delay

The time difference between \( t_t \) and \( t_r \) is computed as delay that includes \( t_w \) and \( p_t \) irrespective of \( \emptyset_r \). The classification and range determination helps to reduce the number of failures in processing requests. Therefore, unnecessary wait time or carry forward of \( \emptyset \), is not required in RTOS. For the shared offloading, the processing is performed by identifying \( y_r(q) \) and in independent offloading \( y_r \) is identified for assigning \( p_c \) such that objective (1b) is satisfied. The basis functions \( B_i(\emptyset) \) and \( B_i(r_p) \) is prominent in determining the \( Y_{r_p}, \frac{t_r}{p_t} \) and \( Y_0 \) as classified using the multivariate spline. Therefore, \( \emptyset_r \) is operated using normal splines as in Figure 2 (a) and 2 (b) [based on range] and modified splines \( p_c, p_t \) as per \( \Delta \) identification recommended by the knowledge base of the CPS. This differential processing using CPS and cloud resources retains the processing and response time of the \( \emptyset_r \), reducing the delay [Refer Figure 6 (a) - 6 (d)].

4.4 Analysis of Response Time

Table 3 Response Time with respect to the Varying Traffic Flows

<table>
<thead>
<tr>
<th>Traffic Flows</th>
<th>DTOS-LBBD</th>
<th>APS-QL</th>
<th>ATO</th>
<th>RTOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>8.85</td>
<td>8.02</td>
<td>6.07</td>
<td>5.12</td>
</tr>
<tr>
<td>400</td>
<td>8.96</td>
<td>8.14</td>
<td>6.19</td>
<td>5.11</td>
</tr>
<tr>
<td>600</td>
<td>9.05</td>
<td>8.22</td>
<td>7.81</td>
<td>6.11</td>
</tr>
<tr>
<td>800</td>
<td>9.15</td>
<td>8.59</td>
<td>6.91</td>
<td>6.19</td>
</tr>
<tr>
<td>1000</td>
<td>9.49</td>
<td>8.78</td>
<td>7.33</td>
<td>6.9</td>
</tr>
</tbody>
</table>
In the proposed RTOS, the response time is confined within \( t_r \) by classifying the range for \( \varPhi_r \). The \( \tau_d \) is differentiated for independent and shared offloading on the basis of the splines and its multivariate. In the independent offloading process, \( \gamma_r \) is used to determine the time lapse between \( t_r \) and \( t_s \) such that the processing time is confined. On the other hand, in the shared offloading process, the actual time of \( t_r \) is confined within the \( t_v \) by identifying \( \gamma_r(q) \) and based on the recommendations of the CPS. The \( \varPhi_r \) that is excess is offloaded to the \( m \) in the increasing order of \( t_s \). Therefore, the time lapse between two or more concurrent \( \varphi_r \) in either of the offloading process is restricted with \( t_v \), reducing the response time. Identifying \( \Delta \) in both the cases adds to the minimization of \( t_r \). The comparative tabulation of response time is presented in Table 3. Similarly, in Table 4 (a) and 4 (b), the comparative analysis of processing time and Processing ratio, and average delay is presented.

Table 4 (a) Processing Time for Varying Users

<table>
<thead>
<tr>
<th>Metric</th>
<th>Users</th>
<th>DTOS-LBBD</th>
<th>APS-QL</th>
<th>ATO</th>
<th>RTOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing Time (ms)</td>
<td>200</td>
<td>5535.162</td>
<td>4998.978</td>
<td>4097.115</td>
<td>2152.443</td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>5917.684</td>
<td>5058.718</td>
<td>4125.357</td>
<td>2264.468</td>
</tr>
<tr>
<td></td>
<td>600</td>
<td>5959.517</td>
<td>5182.12</td>
<td>4322.801</td>
<td>2635.795</td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>6946.169</td>
<td>5445.171</td>
<td>4404.422</td>
<td>3955.88</td>
</tr>
</tbody>
</table>

Table 4 (b) Average Delay for varying Traffic Flows

<table>
<thead>
<tr>
<th>Traffic Flows</th>
<th>DTOS-LBBD</th>
<th>APS-QL</th>
<th>ATO</th>
<th>RTOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>8.85</td>
<td>8.02</td>
<td>6.07</td>
<td>5.12</td>
</tr>
<tr>
<td>400</td>
<td>8.96</td>
<td>8.14</td>
<td>6.19</td>
<td>5.11</td>
</tr>
<tr>
<td>600</td>
<td>9.05</td>
<td>8.22</td>
<td>7.81</td>
<td>6.11</td>
</tr>
<tr>
<td>800</td>
<td>9.15</td>
<td>8.59</td>
<td>6.91</td>
<td>6.19</td>
</tr>
<tr>
<td>1000</td>
<td>9.49</td>
<td>8.78</td>
<td>7.33</td>
<td>6.9</td>
</tr>
</tbody>
</table>

V. CONCLUSION

This manuscript discusses response-aware traffic offloading scheme for improving the request processing rate of user-centric IoT paradigm. The proposed scheme initially classifies the traffic for independent and shared offloading to reduce the processing failures. Offloading process is supported by multivariate spline regression learning model for identifying the time-lapse to assign appropriate service providers to confine additional processing time and delay. Both the offloading process is monitored and supported using the computational recommendations of the CPS coupled with the IoT-Cloud architecture. This helps to improve the processing ratio if the requests and reduce processing time and delay for varying user density and traffic flows.

REFERENCES


