Multi-Agent System for Resource Adaptation in IoT to Monitor Leakage in Irrigation Tanks

Daneshwari I. Hatti

BLDEA's V P Dr P G Halakatti College of Engineering and Technology

Ashok Veerappa Sutagundar (✉ sutagundar@gmail.com)

Basaveshwar Engineering College <https://orcid.org/0000-0003-1494-7150>

Research Article

Keywords: Fuzzy logic, Reinforcement Learning, Multi-Agent, Adaptation, cost, energy consumption, latency, Ifogsim

Posted Date: July 20th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-709807/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Multi-Agent System for Resource Adaptation in IoT to Monitor Leakage in Irrigation Tanks

Daneshwari I. Hatti¹, Ashok V. Sutagundar^{2*}

¹Department of Electronics and Communication, V.P. Dr. P.G.H College of Engineering and Technology,
Vijayapur, India

¹daneshwari_hatti@yahoo.co.in

^{2*}Department of Electronics and Communication, Basaveshwar Engineering College, Bagalkot, India

²sutagundar@gmail.com

Abstract - The sporadic changes in the requirements by the end users has led to the problems in managing the resources of IoT devices. The problem of managing the heterogeneous requests with the available resources in view of ensuring Quality of service (QoS) to end users is challenging issue. The proposed model inculcates the adaptation policy for adapting the resources to fulfill the requirements of the user. The dynamic changes in the environment are handled by Reinforcement learning model with Fuzzy Interference system to apply the policy. Periodically monitoring of irrigation tank to alert the flow of water above Full Tank Level (FTL) by Reinforcement learning agent, prioritization of tasks (requests) by Fuzzy logic is performed. It is done by interacting through agency, providing video conferencing or video calling facility to the user based on availability of the user resources. It also adapts according to communication and computational resources. The proposed model is simulated to monitor and control the leakage in tank. It helps to remotely control the leakages in irrigation tanks/ bridges through Multi-Agent Fuzzy Q learning model. It focuses on adapting the resources of the devices and the action considering the user resources. The algorithm is simulated in Ifogsim and python and performance are evaluated in terms of resource cost, latency, execution time, energy consumption and network usage.

Keywords. Fuzzy logic, Reinforcement Learning, Multi-Agent, Adaptation, cost, energy consumption, latency, Ifogsim

1 Introduction

IoT is emerging technology applied in field of industries, agriculture, irrigation sectors for enabling the communication through Internet with reduced privation. The devices of IoT have capability of sensing, processing, transmitting sensed information to avoid hitches in several sectors. As the devices have marginal resources for computation of real time applications, it takes assistance from Cloud. A real time application leans on resources offering huge computation and communication resources. Increased demand of users increases latency and communication traffic across cloud. Overwhelm the traffic and latency Fog computing, Edge computing [1] has been acquainted. Fog computing offers the virtualization of resources at the edge devices. Fog computing nodes still persists with less resource compared to cloud. Resource management at fog nodes is pivotal for managing the plenty requests from the users. Since the resources of the devices are about to change due to the change in workload. Supervision of resources is utmost important to leverage the services meeting the Quality of Service. The intelligence of IoT devices has led to use in different sectors for accomplishing the tasks in reduced time and losses. Drastic changes in the environment, monitoring the environment conditions is essential for reducing the extent of disaster.

IoT plays vital role in managing the disasters in Industries, forests, residential areas by predicting the consequences happening due to change in environment. Devices of IoT sense the information, processed for predicting the future and alerting the people to take measures. Some of the examples are cyclones, floods, land sliding etc. In these cases, the information sensed by the devices has to be processed and act accordingly with aid of fog devices and cloud. The scarce resources at the IoT layer are efficiently managed by processing at the fog nodes to facilitate human being and other living creatures to save their life. Adaptation to the environment and managing the resources is imperative in view of saving the lives of people. Physical and Network resources adaptation is utmost important for regulating the environment.

The main objective of this work is to monitor the rainfall in areas of bridges, tanks located and avoiding the effects due to leakage in tanks and bridges. In this work Irrigation tanks and tank bunds are used interchangeably. The tanks and bridges are constructed to irrigate the land. In rainy season, monitoring of these tanks, bridges are necessary to avoid the leakages and reduce losses to the farmers. As the tanks constructed in the location covers nearly 40 acres of land for irrigation, if any leakage persists tanks will breach inducing risks to farmers and people located in that residential area. The proposed work detects the leakage in tank bunds and bridges using IoT. The intelligence in the devices using Reinforcement learning and Fuzzy logic is embedded for deciding the application module to be used by the user. Agent technology for allocation of network resources by various application modules based on environment is done. Adaptation of resources is done according to the changes in the environment for reducing the risks of farmers and living creatures in that surrounding area. The resources of fog nodes and users are effectively used for achieving the adaptation goal that is better QoS.

The main contributions of this work are (i) Multiagent approach for communicating between the devices/cameras, fog devices and users. (ii) Resource monitoring prior to processing of requests is performed for increasing lifetime of camera and users. (iii) Adaptation of network resources to handle huge number of requests by the user (iv) Adaptation of resources at device/camera (v) Adaptation of actions to meet the resource requirement by considering the resource availability of the user is performed.

Rest of the paper is organized as follows. Few works highlighting the adaptation of resources for healthcare applications, supply chain management is addressed in section 2. In section 3 processing of data, resource adaptation, resource allocation using reinforcement learning and Fuzzy logic is discussed using example scenario. In section 4 simulation procedure is explained. Section 5 presents the results. Section 6 concludes the work.

2 Related Work

This section describes few works related to adaptation in Cloud, fog and IoT. In [2] authors discussed Self-adaptive system (SAS) which automatically modify itself in response to changes in its operating environment. Adaptation logic based on model-based, architecture-based, reflection, programming

paradigms, control theory, service-oriented, agent-based, nature-inspired, formal modeling and verification, learning and requirements-oriented is surveyed.

In [3] authors have investigated the problem regarding IoT fleets adaptation and proposed a framework for developers to build adaptable applications. Authors [4] have presented a conceptual framework for IoT Architectural Self-adaptation (IAS). The approach facilitates architectural adaptation by correlating it with autonomic and functional control elements. For self-adaptation, the system has to adapt for the changes in network environment is facilitated by Reinforcement learning.

Reinforcement learning (RL) is form of machine learning algorithm [5]. It consists of agent that interacts with environment and learns the strategies through reward-based mechanism. Cumulative reward decides the action for particular state. A model free and model-based approach for managing IoT applications due to their changing nature of requirement is discussed. RL has a characteristic of adapting to the changes in the environment and perform the action with best reward. It relates state, action and reward for any situation. In [6] RL has been applied for adaptive dynamic provisioning to per hop in differentiated service networks assuring Quality of Service (QoS). In [7] RL presents ambient intelligence in Intelligent IoT (IIoT) systems embedding agents offering a class of sequential experience for processing sensory data in order to produce reaction control decisions for several IoT applications.

In [8] RL framework is applied for bandwidth estimation and congestion control in real-time audio/video communication ensuring better Quality of Experience (QoE). RL algorithm is applied for allocating resources in IoT to assure QoE [9]. Fuzzy logic is a simple method for arriving at a conclusion based on vague, ambiguous, imprecise, noisy, or missing input information similar to human reasoning. Fuzzy employs set of rules for deciding the action for the input data. RL and Fuzzy logic based multi-objective routing algorithm for Flying ad hoc networks (FANET) is proposed in [10].

In [11] authors have applied fuzzy logic for updating the states and actions of Q learning variables in monitoring the traffic Flow and Pedestrian Number Estimated from Visual Information by controlling the Traffic Light. Due to increased IoT devices, requests are processed at fog devices and communicate to the end users.

In [12] authors have used RL and Fuzzy in fog computing environment for healthcare applications to reduce latency. In [13] authors have reviewed artificial intelligence and machine learning usage in smart home, healthcare, environmental monitoring, smart transportation, supply chain management and security. Authors [14] have improved the efficiency of applications and services by combining Reinforcement Learning with adaptation mechanisms.

The advantages of RL and fuzzy logic has led to design proposed model for monitoring the Irrigation tanks (Tank bund), bridges and barrages continuously to avoid disasters. RL having capability of interacting environment applies policy for gaining reward in accordance with the decision obtained from Fuzzy logic, as fuzzy logic helps in minimizing the states [15]. The decision for scheduling the requests by adapting the network resource to the changing demand of tank bund and user in view of ensuring better QoS is carried out. In this work multi-agent system along with learning model for fulfilling requirements of the user is proposed in order to control the leakage in irrigation tanks.

3 Proposed Work

The proposed scheme is applied for managing the irrigation tank leakages, overflow of water in barrages, minor and major bridges through video conferencing, analyzing the CCTV footages or by audio call/messages to avoid the failure in the system. In this work water level sensors are deployed at the tanks, minor bridges and major bridges. Level of water is sensed by the IoT sensors, processed by the high-end cameras and send the processed requests to the fog devices. The fog device chooses the application modules, offer the service to the end users with reduced risk of damage. In rainy season there is increased flow of water to the tanks, bridges, barrages and dams. Adaptation to the change in climate and water flow in tank bunds is key factor for reducing the risks of life. For any equipment, buildings or devices have their limitation to withstand the changes. Hence before it explodes the end user is provided with required network resources for monitoring the leakages/damages and controlling the same.

Resource adaptation is performed by reinforcement learning agent and application module is decided by the fuzzy interference engine meeting user requirement. The adaptation mechanism is fulfilled by the fog devices located at the edge devices in regulating the resource starvation of devices. Adaptation according to the requests, emergency level, resources availability at device and user is performed through Fuzzy Reinforcement Learning Adaptation Algorithm (FRLAA). Adaptation policy is done by multiagent as it flexible in communicating among different IoT devices and Fog devices. The preceding section

discusses network environment and proposed adaptation policy for adapting the resources at device and user end.

3.1 Network Environment

The proposed network model shown in Fig. 1 constitutes three layers namely IoT, Fog and Cloud. IoT sensors/camera are embedded with storage, processor and communication unit. It has limited resources for the processing of huge data sensed hence it insists cloud to perform the computation and communicate to increase the lifetime of sensor. Energy of deployed sensor should be conserved by transmitting the sensed data to the edge devices with reduced bandwidth compared to cloud. Fog layer composes router, access points, camera and switches. Fog devices have extra resources compared to IoT devices but comparatively less than Cloud resources. Cloud incorporates high end servers and storage units, processes huge requests and demands. Communication among the devices for adapting to the heterogenous requests is fulfilled by various agency.

[Fig. 1 Network Model]

3.1.1 Model

In this section Multiagent based RL and FLC for adapting resources to the dynamic environment is discussed. The proposed work ensures resource adaptation as per the need of users with assured QoS. It offloads the processing of sensed data to fog device for adapting with change in resources at devices/cameras. It is performed at two levels that is pre-processing of sensed data to avoid communication traffic and resource adaptation for various heterogeneous requests. Dynamic changing of environment is monitored by Reinforcement learning (RL) and decision is forced by Fuzzy Logic Controller (FLC) [16]. Agent approach[17] [18] helps in interaction among several devices, as agents can communicate with heterogeneous devices.

Reinforcement Learning (RL) mechanism which is type of machine learning comprising software agent for interaction with environment by exploring and resulting in action for cumulative reward. It has feedback loop that operates to optimize reward for various action. RL does not have experience it gains by several set of action for current situation. RL uses Q learning wherein a Q-table or matrix is created while performing the Q-learning. The table follows the state and action pair (s, a) and initializes the values to zero. After each action, the table is updated, the q-values are stored within the table. The RL agent uses this Q-table as a reference table to select the best action based on the q-values[19].

The state action table for every response is updated using equation 1 [20] and used for similar requests.

$$Q(S, A) = Q(s, a) + \alpha * (r + \gamma * \max Q(s+1, a) - Q(s, a)) \dots\dots\dots(1)$$

Where Q (S, A)- update new state-action pair

Q(s,a)- Current state-action pair

α -Learning rate

γ - discount factor

r- reward

Fig. 2 shows the elements and working principle of RL.

[Fig. 2 Reinforcement Learning]

Fig. 2 shows the RL mechanism comprising agent, environment, state, action, reward and policy. Agent collects the information from the environment, takes action according to the policy for current state and obtains the reward. Multi-agent in RL is used for performing multiple functions namely (i) pre-processing the data (ii) priority assigning (iii) Fuzzy logic controller (FLC) for decision making in choosing the actions and applying policy of RL and (iv) Adaptation for allocating the resources at fog layer. The cumulative reward obtained after the action is factor for ensuring QoS. RL creates Q matrix constituting state action pair for set of states, gains experience by exploring and exploiting the space and is trained for further similar states. In Fig. 2, agent 1 classifies the sensed data and analyses the user demand by equation 2, assign priority and decides whether to process at fog or cloud based on latency and cost. Agent 2 adopts FLC, analyses the input processed data and decides the required resources for executing the task. Agent 3 searches for best fog device adapting to the requirement sent by agent 2. RL gain experience by exploiting several states in achieving the adaptation of resources for delivering services to tasks with better QoS.

Fuzzy Logic Controller (FLC). It is a type of decision maker for handling vague inputs. It has a capability of decision making similar to human that is by framing set of rules [21]. Several inputs are considered at an interval and appropriate action is obtained [22]. In this work Mamdani fuzzy method is preferred. It includes fuzzifier, knowledge base and defuzzifier. Fuzzifier represents imprecise information by degree of membership function and processes based on several rules existing in Knowledge base followed by defuzzifier to convert in to crisp output. FLC is represented in Fig. 3. In Fig. 3 Fuzzifier fuzzifies the inputs and Fuzzy inference engine applies set of rules for finding appropriate decision. The individual output of several rules is aggregated and defuzzified with center of gravity method. The defuzzified output decides the action to be taken for ensuring QoS to the user. FLC decides the resources required for performing the tasks considering processed data, number of requests, priority, type of application, task requirement and user resources availability. It results in output deciding the quality of video, duration of video, number of requests to be processed meeting the user requirement to assure better QoS and increased resource utilization. FLC is applied for controlling the leakage in tanks through video streaming. User controls the leakage of tank and avoids disaster.

[Fig. 3 Fuzzy Logic Controller]

The adaptation is done in two cases that is (i) Adaptation of resources for processing multiple requests sensed by the sensor deployed at tank bunds. It is done by monitoring the network resources at devices, fog and cloud. (ii) Adaptation of network resources based on the level of emergency and user resources availability is performed.

For the first case sensor senses the water level in tank bunds in terms of FTL and MWL. If the water is above the FTL sensor sends the request for processing. RL updates the current state and finds the action. The action is decided by considering input parameters such as number of requests, tank level, location id, video quality, duration of video, network speed of user and battery level of user. FLC performs an action followed by the update of Q table with new state-action pair and value as shown in Fig. 4. FLC reduces the number of states of RL. For the current state the Fuzzy Interference System (FIS) finds an action and the table is updated. The reward is calculated for the action. If reward is positive for the action taken then the same action is initiated for the upcoming same state and it estimates new action if not available.

[Fig. 4 Fuzzy RL model]

Adaptation policy is employed at three layers by adapting to changes in number of requests, tank level, communication and computational resources. It is performed to ensure reduced latency, bandwidth and energy consumption. In this section proposed work is discussed with example scenario, RL and fuzzy approach for network resource adaptation and agency is discussed.

3.1.2 Agency

The proposed algorithm FRLAA comprises three agencies namely Device, Fog Agency and User agency. The communication among three layers for adaptation and allocation of resources is performed by these agencies. Agents are employed for collecting the sensed data from IoT sensors, transmitting the processed

data to Fuzzy, applying adaptation policy for allocating resources to users, computing rewards and finding optimal action. For interacting between IoT device, Fog device and Cloud layer, multiagent namely DTA, VTA, RMA, RLA, DMA, RA, URMA is created. The processed data is sent by DTA to DMA fuzzy interference engine for taking action for the current state. The policy is employed by RLA for updating the network resources in provisioning to current state. RL and Fuzzy Logic controller (FLC) is simulated at device, fog layer and cloud layer based on the resource availability at user.

a. Device agency

It constitutes static and many mobile agents for monitoring the tank/bridges level and controlling leakages in tanks. It has Sensor Knowledge Base (SKB), Data Transfer Agent (DTA), Video Transfer Agent (VTA) and Device Resource Manager Agent (DRMA). Fig. 5 shows the device agency. DTA is mobile agent, created by sensor/camera to monitor the tank level and when it is above threshold that is above FTL, the DTA communicates with DRMA for estimating the resource availability at device end. If resources are adequate the Reinforcement Learning Agent (RLA), Reward Agent (RA), Decision Maker Agent (DMA) are created at device by DRMA. RLA updates the SKB periodically, processes the new requests. RLA employs RL for updating the new state and FLC for choosing the policy to take action for the state detected by RL. The policy or decision is done by DMA for taking proper action and the action is communicated by VTA through DRMA. Processing of video according to the action obtained is done at DRMA then communicated through VTA.

If device does not have sufficient resources, then DTA interacts with FRMA of fog agency to adapt the adaptation policy at fog layer instead of processing at device agency.

[Fig. 5 Device Agency]

b. Fog agency

It consists of a Fog Knowledge Base (FKB), Fog Resource Manager Agent (FRMA) and Resource Adaptation Agent (RAA). RAA is the static agent that adapts to the requirements of the user request. Fig. 6 depicts fog agency. It checks whether the required resource is at fog device then creates RLA, DMA and RA. The RLA is reinforcement learning agent learns and updates the Q table constituting requests and action taken. Agent arriving from user that is User Information Transfer Agent (UITA) sends the information about the user requirement and availability of resources to FRMA through RAA. DMA fetches information from FRMA and applies Fuzzy Logic Controller for finding appropriate action required for the current state and is updated by FKB. If the resource is available then the DMA sends request to VTA which is migrating to user agency for sending the required quality video to assure the user satisfaction. If user is satisfied RA updates the reward and communicate with RLA. If resources are not available at fog devices RAA communicates with cloud and send the request to process at cloud through agent. FRMA is a static agent residing in each fog device of a fog colony. Each RLA monitors the tank and FRMA monitors the resources periodically and updates FKB with current resource availability and resource configuration.

[Fig. 6 Fog Agency]

c. User agency

It includes User Information Transfer Agent (UITA) and User Resource Manager agent (URMA) that is depicted in Fig. 7 UITA is mobile agent created by user for sending availability of resources, latency tolerable and capability of handling the requests to the FRMA or DRMA. URMA updates the knowledge base of user and interacts with UITA for transfer of user information to the FRMA or DRMA based on availability of resources at fog and device. If user has inadequate resources for handling further requests, then such hard deadline requests are processed at nearest fog device.

[Fig. 7 User Agency]

The interaction of various agency is illustrated in Fig. 8. Fig. 8 shows the sequence of interactions among the agents of the agency. The numbers on the arcs specify the type of actions. Agent interaction sequence is described below.

1. Sensors sense the tank level

2. Sensed data is sent to DRMA.
3. DRMA classifies data and decides whether to be processed, if tank level >FTL then DRMA fetches information of sensor resources from SKB.
4. If device is having adequate resources, then DRMA creates RLA.
5. DRMA creates DMA for adaptation of resources by monitoring user resources availability.
- 5a. URMA creates UITA for sending user information to DMA.
6. RA is created by DRMA through RLA. RLA applies RL for updating Q table.
7. DMA applies FLC for applying policy to RL and creates VTA for sending video.
- 7a. Based on DMA decision, VTA sends video to URMA at user agency.
- 7b. Based on decision the URMA monitors the tank and RA is estimated. RA helps in updating of Q table to find action for various situations.
8. DRMA updates SKB and processes further requests. If resources are not available at Device, DTA is created by DRMA.
9. DTA communicates with FRMA.
10. FRMA at the fog agency decides to create agents.
11. RLA is created by FRMA
12. DMA for taking applying policy to RL.
- 12a. URMA send UITA for forwarding user resources availability to the DMA.
13. RA is created to estimate the reward for actions taken by DMA to adapt the available resources.
14. URMA updates the availability of user resources.
15. If adequate resources are not available at user, then processing is forwarded to nearest fog device.
16. If resources are not sufficient at user and fog device then request is forwarded to cloud by FRMA.
17. The VTA then communicates with URMA for establishing communication between tank and user for monitoring leakages and managing tanks through video streaming.

[Fig. 8 Interaction among Agency for adaptation of resources]

3.2 Algorithm

The proposed multiagent based Fuzzy Reinforcement adaptation algorithm is described in the below algorithm. Adaptation policy with multiple agencies for adapting the processing of requests at device, fog and cloud based on availability of resources is done. Adaptation and allocation of resources to the user based on availability of resources at user end for monitoring the leakages in tanks with reduced latency, energy consumption and network usage is discussed.

Nomenclature. No of sensors ‘si’; Sensed Input ‘sdi’; no of camera ‘ci’; Video quality ‘vi’; User requirement ‘ui= {li, ei, ci}’; Executed tasks requests ‘ti’;

Inputs. si; sdi; ui= {li, ei, ci}, vi;

Output. ti;

Begin

1. Deploy sensors/cameras near tanks/bridges
2. Create cluster of nearest sensors located on tanks in area of radius ‘n’ kms.
3. Update respective DRMA of the clusters with tank level in terms of FTL, MWL.
4. **For** i=1 to n
 begin
5. If (sdi ≥ FTL)
6. DTA communicates with DRMA and offloads the request processing based on equation 2 by taking in to consideration of resource availability (Bandwidth) at devices ‘xi’, user resource availability (BW) ‘x’ and CPU of device ‘yi’ and CPU of user ‘y’

$$R = BW * \left[\frac{e^{(x_i - x)}}{e^{(x_i - x)} + 1} \right] + CPU * \left[\frac{e^{(y_i - y)}}{e^{(y_i - y)} + 1} \right] \dots\dots\dots (2)$$

End
End
7. **If** (*res_dev* > *res_req*)
DRMA creates RLA, DMA, RA
Goto step 8
Else if process at fog agency
DRMA interacts with FRMA and creates RLA, DMA, RA
Else process at Cloud
End
8. *RLA, DMA, RA agents are created by respective Resource Manager*
8a. *RLA applies RL to monitor periodically and learn based on the policy applied by DMA*
8b. *DMA interacts with URMA and applies FLC*
8c. *The decision obtained from DMA is communicated through VTA*
8d. *RA calculates the reward and updates the Q table, state-action pair Q (S, A) using equation 1 and reward is calculated using equation 3*

$$Q(S, A) = Q(s, a) + \alpha * (r + \gamma * \max Q(s+1, a) - Q(s, a)) \dots\dots\dots (1)$$

$$RA = \begin{cases} 1 & \text{user_req} = \text{VTA_resp} \\ -1 & \text{user_req} \neq \text{VTA_resp} \end{cases} \dots\dots\dots (3)$$

9. **If** (*user_res* < *req_res*)
Offload the URMA on available nearest Fog device
Else
Process at cloud
End
10. *Compute energy consumption, delay, resource cost, network usage, no. of requests processed for different configuration used for adapting to the different scenarios.*
End

3.2.1 Example Scenario

The tanks are designed and constructed for the welfare of the people. The irrigation tanks are of different heights designed to afford irrigation for several acres of land. Tanks have different water level to be monitored and warned when it reaches the maximum height. Irrigation tank cross sectional view is shown in Fig. 9. It constitutes three water levels Top Bund Level (TBL), Maximum Water Level (MWL), Full Tank Level (FTL) and it includes catchment area. Tank are constructed with different heights and is depicted in Fig. 9. Monitoring of water level is essential because if failed to monitor and intimate the staff, tank breaches due to overflow of water causing destruction in living creatures and agriculture lands.

[**Fig. 9** Cross section view of Irrigation tank (Tank Bund)]

In a particular taluka of any district, tanks are constructed for benefit of human beings to cultivate the barren land and increase ground water table. The problem rises in rainy season due to heavy rainfall of approximately up to 400-600mm in concerned catchment areas of the tanks or near to tanks leading to heavy flow of water in tanks. When the tank is full it has a catchment area for flow of excess water that is above Full Tank Level (FTL).

The overflow of water drastically increases the chance of breaching the tank resulting in damage and inundation of water. Overcome this problem, tanks water level has to be monitored regularly and act immediately. In each taluka minimum of 40 tanks are existing and looked after by concerned irrigation department depending upon irrigable area. Consider one situation where heavy rainfall occurs across at least 3-4 tanks resulting in heavy water flow, it is tedious to report for every location in short duration to monitor the tanks by individual maintenance staff. Above FTL the water flowing through catchment area will have reduced risk. If excess water is not allowed through tanks, the flow from the concerned tanks of the catchment area results in reaching MWL of the tank. To maintain this proper watch and ward is essential.

In the proposed model, rising of water level above FTL, alert to the user is initiated. The user periodically monitors the raise of water level and takes appropriate action. If water is approaching MWL slightly from FTL then possibility of leakage increases (If tanks are not properly maintained, before reaching FTL the chance of leakage may be).

To solve this issue the proposed model monitors tanks and bridges, adapts and allocates network resources for handling the situation remotely. It is illustrated in example scenario shown in Fig. 10. It consists of sensors (1-50), GPS for finding location of tanks and bridges, fog devices, cloud and Agents. The operation is as follows.

- A. detection of water level at regular intervals.
- B. transmits the sensed data, location id, priority and user requirement
- C. processing by RLA
- D. priority assigning and forwarding requests to process at fog and cloud based on user requirement by DMA
- E. requests are forwarded to cloud and fog for processing if resources are inadequate.
- F. action is decided and allocation of resources by DMA to URMA through VTA
- G. Q matrix is updated based on reward obtained by RA

Level of water in a tank is sensed by sensor along with height of tank, location id, approximate count of surrounding living creatures and resource requirement for accomplishing the task to avoid any discrepancies. Similarly, minor and major bridges have guard stones to indicate the raise level of water. The sensors deployed senses and informs the user to take appropriate action immediately to save the life of people. At the same time if multiple requests come to single user, it is difficult to ensure the service in stipulated time. Hence remotely instructions are given to the end users located near the bridges and perform the operation minimizing the delay. In order to provide the services remotely, network resources are to be managed efficiently in less duration to avoid huge loss. Optimal usage of resources for performing the task without violating QoS parameters and to adapt for changing resources of the user is performed by the proposed model and simulated.

The model shown is used as an example to manage water level through the user by adapting to the network resources as shown in Fig. 10. Sensors/ high end cameras are deployed on guard stone of tank; sensor senses the water level and communicates with RL agent. It employs Q learning algorithm preprocess the data, priority of the request is addressed by Fuzzy Interference System (FIS). $U_1, U_2 \dots U_n$ are the users interacting with FIS for exchange of resource information and action to be taken for the sensed state. The action in turn decides the quality of video and duration required for handling present request. FIS communicates with RL and camera for sending the video. Users directly controls the tank through the user having internet connectivity.

[Fig. 10 Adaptation policy for monitoring leakage in irrigation tanks]

[Fig. 11 Application Model of Adaptation mechanism in Ifogsim]

Fig. 11 depicts the application model used for simulating the proposed algorithm in Ifogsim. The application models are water level sensor, data process gateway, fuzzy RL agent, camera and end user. These are created in Ifogsim for communicating, processing the requests. The different application modules are placed on device, fog and cloud by applying the proposed adaptation policy. The simulation of the proposed model is discussed in next section.

4 Simulation

The proposed model is simulated for analyzing the performance of the system with fog and cloud-based scenarios using python and Ifogsim. In this section simulation procedure and performance parameters are discussed.

4.1 Simulation procedure

Simulation steps for simulating the proposed model are as follows.

1. Generate the network environment deployed with sensors and camera and set the parameters.
2. Apply Fuzzy logic Q learning for adapting resources at user and device end.
3. Adaptation is performed at device end for forwarding request to fog and cloud
4. Vary number of request and compare with fog based and cloud-based approach
5. Vary the request apply adaptation policy
6. Compute performance parameters latency, energy consumption, cost, network usage and execution time.

Fig. 12 illustrates the topology implemented in Ifogsim [23] for analyzing the performance of the proposed adaptation policy. Cloud is at level 0, Proxy server at level 1, Fog at level 2 and Mobile at level 3. Mobile is device handles the water level sensor and user request. Fog at level 2 processes the requests in case of inadequate resources at devices. The proposed adaptation policy adapts to the available resources at mobile-0 and mobile-1 and then switches to processing at fog device. The different application modules used in Ifogsim acts as multiagent for performing their job as described in section 3.2.

[Fig. 12 Topology of proposed model in Ifogsim]

[Table 1 Configuration of devices]

[Table 2 Configuration and Specifications]

Table 1 and 2 represents the configuration of devices [24]. Table 3 represents simulation inputs used for simulation of the proposed model.

[Table 3 Simulation Inputs]

Table 4 shows the FIS sample set of rules framed for evaluating the current state and action is decided by FIS. The FIS model is evaluated using python and the defuzzified output is the action obtained for the current state. The FIS model is designed with 40 rules to consider all the cases. The defuzzifier uses center of gravity method to aggregate all the rules.

[Table 4 FIS sample set of rules]

5 Results

In this section results are depicted for Adaptation using Fuzzy Q learning approach [25]. It is simulated for various configurations. Various configuration includes processing at device, fog and cloud layers. Adaptation for processing requests at device, fog and cloud is done by Fuzzy Q learning model.

The set of rules are framed for deciding the priority of request or task based on the water level detected by sensor, rules also decide the quality of video that is 144p, 240p, 360, 480p, 720p for duration 5-15 min. The output of fuzzy is obtained by considering number of requests, emergency level, user resource availability and resource at devices. The RL learns and Q table is updated after 100 iterations and finally action is updated. Reward is obtained for the action taken in terms of -1 and +1. If it is '-1' then for next time the action is not taken for this same state, it updates to new state-action pair. The processing of such requests is measured in terms of performance measures execution time, delay or latency in milliseconds, energy consumption, resource cost and network usage.

The processing is done at cloud, fog and device for the incoming requests. The requests processed at cloud consumes more energy, network resources and increases resource cost, latency and execution time

compared to fog and cloud. The number of incoming requests is processed completely at cloud but all the requests are not processed at fog and cloud due to insufficient resources. The adaptation policy offloads the complete or partial request to fog and cloud in case of scarce resources at device.

Fig. 13 shows the comparison of processing at cloud, fog and device with 60 incoming requests. Compared to cloud and fog the processing at device incurs less delay. Fig. 14 shows the energy consumption, cost and network usage for the requests processed at cloud, fog and IoT device. Processing at device incurs less energy consumption, resource cost and consumes less network resources.

[Fig. 13 Comparison of processing at Cloud, fog and device]

[Fig. 14 Performance Measures]

Fig. 15 shows the number of processed requests at IoT device, fog and cloud with adaptation policy. For 10 requests IoT device has executed but for 30 requests few tasks are offloaded to fog device due to scarce resources at device. For 60 requests adaptation policy has applied to offload the tasks based on the availability of resources at device, fog and cloud. Fig. 16 shows the comparison of completed requests v/s incoming requests with adaptation and without adaptation. If incoming requests are 10, with adaptation policy all 10 requests are completed the execution whereas without adaptation only 6 requests are executed.

[Fig. 15 No. of requests execution with adaptation policy]

[Fig. 16 Comparison of processing no. of requests with adaptation and without adaptation]

Proposed algorithm with Adaptation policy results in reduced delay compared to traditional approach that is without adaptation. Fig. 17 shows the delay (ms) that application loop delay occurred for processing the incoming request in particular location. Delay increases as the number of requests are increasing but comparatively it is less than traditional approach.

[Fig. 17 Delay for processing no. of requests with adaptation and without adaptation]

Proposed algorithm with Adaptation policy results in reduced energy consumption as compared to traditional approach that is without adaptation. Fig. 18 shows the energy consumption (MJ) resulted for processing the incoming request in particular location. Energy consumption increases as the number of requests increases. Since the device having scarce resources cannot afford to process huge requests forwards to cloud resulting in increased energy consumption. Due to adaptation policy few requests are processed at device and then sent to cloud hence energy consumption is reduced compared to traditional approach.

[Fig. 18 Energy Consumption for processing no. of requests with adaptation and without adaptation]

Fig. 19 shows the comparison of proposed work with QoS aware Resource Management, proposed work performs better in terms of delay and energy consumption. The simulated work shows the result for two configuration that is IoT-Fog and IoT-cloud approach. IoT- Fog performs better with reduced delay compared to IoT-Cloud. Proposed IoT-Fog configuration performs better than the QoS aware RM in IoT work with reduced energy consumption (MJ).

[Fig. 19 Comparison of proposed work with QoS aware Resource management in Fog]

6 Conclusion

This paper depicts the multi-Agent based Fuzzy Q learning model for adaptation of computational resources to meet the user requirement. Adaptation of resources according to the availability at the IoT device is also performed. Fuzzy Q learning model has added advantage to adapt the resources to monitor the leakage in tanks and control through video conferencing. This work has addressed adaptation of resources by observing both user and IoT devices and accordingly processing is offloaded to fog device and cloud. The Multiagent approach along with learning model resulted in reduced energy consumption compared to the existing work. The proposed algorithm is simulated for different architectural framework centralized and distributed. To evaluate, the two architectures are simulated and compared with performance measures such as resource cost, execution time, resource cost and latency. The distributed approach performs better compared to centralized. The proposed work is simulated with and without adaptation mechanism. By employing adaptation mechanism considering the available resources across user and devices end, the request processed are tabulated. The further improvement is to process a huge number of requests considering other performance measures and implement on testbed.

Funding

Not applicable

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Adi E, Anwar A, Baig Z, Zeadally S (2020) Machine learning and data analytics for the IoT. *Neural Comput Appl* 32:16205–16233 . <https://doi.org/10.1007/s00521-020-04874-y>
2. Krupitzer C, Roth FM, VanSyckel S, Schiele G, Becker C (2015) A survey on engineering approaches for self-adaptive systems. *Pervasive Mob Comput* 17:184–206 . <https://doi.org/10.1016/j.pmcj.2014.09.009>
3. Achtaich A, Souissi N, Mazo R, Salinesi C, Roudies O (2018) Designing a Framework for Smart IoT Adaptations. In: Belqasmi F, Harroud H, Agueh M, Dssouli R, Kamoun F (eds) *Emerging Technologies for Developing Countries*. Springer International Publishing, Cham, pp 57–66
4. Moghaddam MT, Rutten E, Lalanda P, Giraud G (2020) IAS: An IoT Architectural Self-adaptation Framework. In: Jansen A, Malavolta I, Muccini H, Ozkaya I, Zimmermann O (eds) *Software Architecture*. Springer International Publishing, Cham, pp 333–351
5. Dhaya R, Kanthavel R, Algarni F, Jayarajan P, Mahor A (2020) Reinforcement Learning Concepts Ministering Smart City Applications Using IoT. In: Kanagachidambaresan GR, Maheswar R, Manikandan V, Ramakrishnan K (eds) *Internet of Things in Smart Technologies for Sustainable Urban Development*. Springer International Publishing, Cham, pp 19–41
6. Hui TC-K, Tham C-K (2003) Reinforcement learning-based dynamic bandwidth provisioning for quality of service in differentiated services networks. In: *ICON*. pp 507–512
7. Nouredine DB, Krichen M, Mechti S, Nahhal T, Adoni WYH (2020) Towards an Agent-Based Architecture using Deep Reinforcement Learning for Intelligent Internet of Things Applications. pdf
8. Fang J, Ellis M, Li B, Liu S, Hosseinkashi Y, Revow M, Sadovnikov A, Liu Z, Cheng P, Ashok S (2019) Reinforcement learning for bandwidth estimation and congestion control in real-time communications. *ArXiv Prepr ArXiv191202222*
9. Gai K, Qiu M (2018) Optimal resource allocation using reinforcement learning for IoT content-centric services. *Appl Soft Comput* 70:12–21 . <https://doi.org/10.1016/j.asoc.2018.03.056>
10. Yang Q, Jang S-J, Yoo S-J (2020) Q-Learning-Based Fuzzy Logic for Multi-objective Routing Algorithm in Flying Ad Hoc Networks. *Wirel Pers Commun* 113:115–138 . <https://doi.org/10.1007/s11277-020-07181-w>
11. Moghaddam MJ, Hosseini M, Safabakhsh R (2019) Fuzzy Q-Learning Traffic Light Control based on Traffic Flow and Pedestrian Number Estimated from Visual Information. *J Comput Knowl Eng* 2:
12. Shukla S, Hassan MF, Jung LT, Awang A (2019) Architecture for Latency Reduction in Healthcare Internet-of-Things Using Reinforcement Learning and Fuzzy Based Fog Computing. In: Saeed F, Gazem N, Mohammed F, Busalim A (eds) *Recent Trends in Data Science and Soft Computing*. Springer International Publishing, Cham, pp 372–383
13. Zantalis F, Koulouras G, Karabetsos S, Kandris D (2019) A Review of Machine Learning and IoT in Smart Transportation. *Future Internet* 11:94 . <https://doi.org/10.3390/fi11040094>
14. Argerich MF (2018) Learning based Adaptation for Fog and Edge Computing Applications and Services. 65

15. Sun X, Xie Z (2020) Reinforcement Learning-Based Backstepping Control for Container Cranes. In: Math. Probl. Eng. <https://www.hindawi.com/journals/mpe/2020/2548319/>. Accessed 7 Feb 2021
16. Hatti DI, Sutagundar AV (2017) Fuzzy based job classification and resource allocation in IoT. In: 2017 International Conference on Inventive Systems and Control (ICISC). pp 1–4
17. Birje MN, Manvi SS, Prasad B (2006) Resource discovery and allocation in wireless grids by using a multiagent system. *Multiagent Grid Syst* 2:237–252 . <https://doi.org/10.3233/MGS-2006-2303>
18. Sutagundar AV, Manvi SSS, Birje MN (2007) Agent-Based Location Aware Services in Wireless Mobile Networks. In: The Third Advanced International Conference on Telecommunications (AICT'07). IEEE, Morne, Mauritius, pp 20–20
19. Harmon ME, Harmon SS (1997) Reinforcement Learning: A Tutorial.: Defense Technical Information Center, Fort Belvoir, VA
20. Xiong X, Zheng K, Lei L, Hou L (2020) Resource Allocation Based on Deep Reinforcement Learning in IoT Edge Computing. *IEEE J Sel Areas Commun* 38:1133–1146 . <https://doi.org/10.1109/JSAC.2020.2986615>
21. Huang J, Angelov PP, Yin C (2020) Interpretable policies for reinforcement learning by empirical fuzzy sets. *Eng Appl Artif Intell* 91:103559 . <https://doi.org/10.1016/j.engappai.2020.103559>
22. Akinwumi E (2020) Multi-Agent Fuzzy Reinforcement Learning for Autonomous Vehicles. Master of Applied Science, Carleton University
23. Perez Abreu D, Velasquez K, Curado M, Monteiro E (2020) A comparative analysis of simulators for the Cloud to Fog continuum. *Simul Model Pract Theory* 101:102029 . <https://doi.org/10.1016/j.simpat.2019.102029>
24. Gupta H, Dastjerdi AV, Ghosh SK, Buyya R (2016) iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in Internet of Things, Edge and Fog Computing Environments. 22
25. Arabnejad H, Pahl C, Jamshidi P, Estrada G (2017) A Comparison of Reinforcement Learning Techniques for Fuzzy Cloud Auto-Scaling. *ArXiv170507114 Cs*

List of Fig.s

Fig. 1. Network Model

Fig. 2. Reinforcement Learning

Fig. 3. Fuzzy Logic Controller

Fig. 4. Fuzzy RL model

Fig. 5. Device Agency

Fig. 6. Fog Agency

Fig. 7. User Agency

Fig. 8. Interaction among Agency for adaptation of resources

Fig. 9. Cross section view of Irrigation tank (Tank Bund)

Fig. 10. Adaptation policy for monitoring leakage in irrigation tanks

Fig. 11. Application Model of Adaptation mechanism in Ifogsim

Fig. 12. Topology of proposed model in Ifogsim

Fig. 13. Performance Measures

Fig. 14. Performance Measures

Fig. 15. No. of requests execution with adaptation policy

Fig. 16. Comparison of processing no. of requests with adaptation and without adaptation

Fig. 17. Delay for processing no. of requests with adaptation and without adaptation

Fig. 18. Energy Consumption for processing no. of requests with adaptation and without adaptation

Fig. 19. Comparison of proposed work with QoS aware Resource management in Fog

List of Tables

Table 1. Configuration of devices

Sl.No	Hardware units of servers of Fog and Cloud	
1	Architecture	X86
2	VM	Xen
3	Operating System	Linux

Table 2. Configuration and Specifications

Name	MIPS	RAM	Up Bw	Down BW	Level	Rate per MIPS	Busy power (MW)	Idle Power (MW)
Cloud	44800	40000	100	10000	0	0.1	1648	1.332
Fog device	2800	4000	10000	10000	1	0.01	107.33	83.44
Gateway	2800	4000	10000	10000	2	0.005	107.33	83.44
IoT Device (camera)	500	1000	10000	10000	3	0.005	87.33	82.44

Table 3. Simulation Inputs

Parameters	Notation	Values
No of sensors	N	1-40
Location Id	Lid	1-5
Energy of sensor node	Ei	1KJ
Tank Level (sensed data)	Sdi	15-100 feet
No of fog devices	Fi	1-10
Battery (%)	Bi	30-90
Network Speed	Ns	512 Kbps-10Mbps
Learning rate	α	0.1
Discount factor	γ	0.9

Table 4. FIS sample set of rules

If tank_level is high and battery is med and no_of_req is med and nw_speed is med then video_qu is high
If tank_level is med and battery is med and no_of_req is med and nw_speed is med then video_qu is med
If tank_level is high and battery is low and no_of_req is high and nw_speed is med then video_qu is low