Journal of Ubiquitous Systems & Pervasive Networks
Volume 15, No. 1 (2021) pp. 17-25

Context-based Reasoning through Fuzzy Logic for Edge Intelligence

Ramin Firouzi [a] *, Rahim Rahmani [a], Theo Kanter [a]

[a] *Department of Computer and Systems Science, Stockholm University, Kista, Sweden, SE-164 07*

Abstract

With the advent of edge computing, the Internet of Things (IoT) environment has the ability to process data locally. The complexity of the context reasoning process can be scattered across several edge nodes that are physically placed at the source of the qualitative information by moving the processing and knowledge inference to the edge of the IoT network. This facilitates the real-time processing of a large range of rich data sources that would be less complex and expensive compare to the traditional centralized cloud system. In this paper, we propose a novel approach to provide low-level intelligence for IoT applications through an IoT edge controller that is leveraging the Fuzzy Logic Controller along with edge computing. This low-level intelligence, together with cloud-based intelligence, forms the distributed IoT intelligence. The proposed controller allows distributed IoT gateway to manage input uncertainties; besides, by interacting with its environment, the learning system can enhance its performance over time, which leads to improving the reliability of the IoT gateway. Therefore, such a controller is able to offer different context-aware reasoning to alleviate the distributed IoT. A simulated smart home scenario has been done to prove the plausibility of the low-level intelligence concerning reducing latency and more accurate prediction through learning experiences at the edge.

Keywords: *Internet of Things (IoT), context-awareness, edge computing, reasoning, type two fuzzy controller*.

1. Introduction

In the forthcoming years, with the continued flourishing of technology— identification, data capture, and processing capabilities—and increasing integration of large-scale data, even small- and medium-scale players in every sector of the industry would be drawn to adopt IoT solutions and services. It means hundreds of billions of things will be connected to the IoT in the near future. Data collection and data contextualization of devices have already been addressed in earlier researches [1–3]. Most of the research has addressed these challenges through the middleware and IoT platforms.

The weakness of these solutions are twofold: first, to address each challenge, one middleware is required (e.g., device management, protocol conversion, context-awareness) and there is no comprehensive middleware solution which is capable of addressing all aspects required by IoT, and, secondly, they are cloud-centric [4].

On the other hand, the vast number of devices which is expected to be connected to IoT lead to building very complex systems and architecture with a huge number of components, such as devices and service and so forth. In order to achieve expected stability and robustness mostly, complex controllers are needed; therefore, complex and uncertain plants usually cannot be encountered with well-known linear approaches. Complex controllers usually lack a straightforward design methodology, and their actual implementation is difficult (if not impossible). Fuzzy logic control is an intelligent technology that allows the conversion from logic statements to a nonlinear mapping based on it [5].

Nevertheless, the traditional approach of using intelligence in the cloud can be inefficient and call for more computational capacity, which is very expensive with the enormous amount of incoming data and huge forthcoming devices. Meanwhile, the advent of edge computing lets us use the computational capacity that is distributed over the network. It means edge computing is promising technology can pave the path for the industry players and research communities to cope with these issues. In this research, first, we introduce a two-level architecture for the adoption of edge computing to enabling distributed intelligence in order to provide intelligence of things by reaping the information of things closer to the devices. Then we propose an IoT gateway controller by leveraging fuzzy logic control to provide low-level intelligence (i.e., edge-intelligence) to the small data at the edge before providing high-level intelligence at the cloud. In order to demonstrate the feasibility of the proposed approach, a smart home application was exploited.

To summarize, our goals are:

- a novel two-level intelligence architecture to provide low-level intelligence closer the devices that reduces latency, optimizes the use of network bandwidth, and offload part of burden from cloud to the edge.
- an implementation of IoT gateway regarding the low-level intelligence in the architecture to demonstrate its performance in the smart home scenario; our results

* Corresponding author.
E-mail: ramin@dsv.su.se
© 2021 International Association for Sharing Knowledge and Sustainability.
DOI: 10.5383/JUSPN.15.01.003

show significant improvements in terms of latency compare to the cloud.

- a fuzzy logic controller as a reasoner module in the gateway that learns and predicts the desired temperature for a cottage based on the location of the owner and data from sensors in the cottage. Results show higher accuracy in comparison to simple ruled based reasoner.
- Setting fuzzy rules to improve efficiency of our proposed method.
- Saving energy. Furthermore, it helps achieving financial savings.

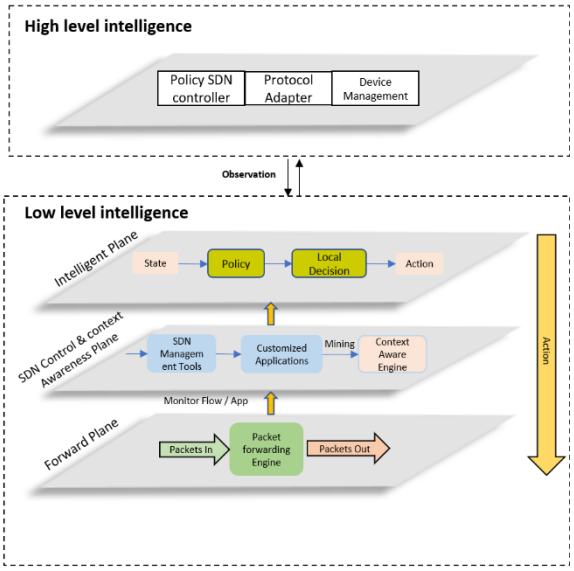

Fig. 1. Distributed intelligent gateway controller in distributed IoT.

The rest of the paper is organized as follows. Section 2, we introduced a prominent topic in this research, and related work with distributed reasoning. In Section 3, we introduced the two-level intelligence and the main characteristics of the proposed approach, autonomic gateway architecture. In Section 4, we focus on performance evaluation. Finally, Section 5 presents conclusions.

2. Related Works

In this section, we will first introduce reasoning then discuss various approaches proposed earlier for distributing reasoning in IoT. As stated in [6], there are two ways to study and build intelligence systems: top-down and bottom-up. The top-down study focuses on intelligent behavior such as thought and reasoning. For computers to simulate the same behavior, people need to figure out what an agent needs to know in order to trigger that behavior and what computational mechanisms could allow them to do to gain new knowledge based on the existing knowledge. Reasoning is one of the intelligent behaviors, which is a top-down method. Reasoning is about making conclusions and deriving new facts that do not exist in the knowledge base. Reasoning entails two essential concepts: knowledge and inference. Reasoning is a process to infer new knowledge based on existing knowledge. The same knowledge could have several different ways to be represented [7].

IoT leads to more challenges for reasoning; for instance, at any step of the data delivery process, from the sensor node to backend knowledge repositories, reasoning can occur. Reasoning latency with large data sets can be physically improved by distributing reasoning tasks [7]. In [8], it is noted that performance of the knowledge system can be increased by distributing of tasks which is due to that distributing of tasks leads to improving problem-solving of capacity and efficiency, expanding the scope of the application—otherwise known as domain—and facilitating implementation by splitting tasks into subtasks. The authors identify that distributed intelligence has advantages under three conditions; first, the data, knowledge, and control must also be distributed physically in addition to logically; second, the cost of communication is much lower than that of problem solution; and, last, system components collaborate with each other to solve the problem. In this paper, all of these conditions have been met.

In other research [9], the authors pointed out other conditions in order to distribute reasoning regarding computational, communication, scalability, and availability advantages of distributed reasoning in dynamic and heterogeneous environments. First, data is highly dynamic and has ambiguous context; second, the amount of data is large compared to the computational capabilities of the IoT nodes; and third, collective intelligence can be achieved by sharing data and reasoning tasks.

In [10], the authors discussed distributed reasoning in multiagent systems (MASs), where distributed software agents make decisions and operate collaboratively to reach some common goals for clients. Badica et al. [11] surveyed Rule-based multiagent reasoning in the field of Ambient intelligence. The drawback of most MAS is that they have been developed for specific environments, support only relatively narrow knowledge domains, and are mostly closed systems using miscellaneous protocols, standards, and interfaces.

Although the papers presented, discuss real use cases for distributing reasoning in IoT, all of them have a drawback they are cloud-based solutions as it is mentioned in *Section 1*; thus, aforementioned proposals straight our vision toward implementing IoT controller in order to distribute the reasoning of the context-aware application over the edge of IoT.

3. Proposed approach

3.1. Distributed Intelligent IoT Controller

A Distributed Intelligent IoT controller, similar to IoT gateway, can help the edge computing by means of utilizing resource-constrained. Given this, we introduce a distributed two-level intelligence with a three-planes logical functionality architecture on low-level intelligence for the edge of the IoT controller, as shown in *Figure 1*. The underlying motivation for developing a two-level intelligence for the edge of the IoT is to offload the burden from the cloud as cloud-centric solutions fail to provide low-latency, which is one of the IoT requirements. To reap value from both edge and cloud, we discuss how and why the intelligent plane can be located closed to the IoT edge devices (i.e., Distributed not centralized). As illustrated in *Figure 1*, the framework consists of two-level intelligence, high-level intelligence, and low-level intelligence.

The cloud controller would provide high-level intelligence [12]. Low-level consists of three planes called the intelligence plane, Software-Defined Networking (SDN) control and Context-awareness plan, and the forward plan. The two planes on the low-level intelligence, Intelligence plan, and SDN control and Context-awareness plane are the main building blocks for the Distributed Intelligent IoT controller. These building blocks are specific for different context-aware applications. In the third

plane, the forwarded plane is the same for all applications. The purpose of the intelligence plane is a mechanism of the learning process, which makes it possible for controller IoT gateway to manage input uncertainties, and the learning system can improve its performance over time by interacting with its environment, and the controller will be able to offer different context-aware reasoning to alleviate the distributed IoT. The SDN control and Context-awareness plane is responsible to the monitoring of communication between the applications and services in the cloud and user devices (i.e., smart city or smart home applications); therefore, the IoT gateways can be managed on the real-time needs and status dynamically and its ability to responds to any context with reasoning based on context-awareness. The forwarding plane is responsible for forwarding data to user devices or Wireless Sensor Networks (WSN). Its operation relies on packet forwarding engine.
Based on three plans, the controller can continually learn and optimize devices and protocols management strategies by interacting with the IoT devices (*Figure 1*).

3.2. Autonomic IoT Gateway

In this paper, to implement low-level intelligence at the edge of the network by the gateway, we introduced three modules, namely device management, protocol adapter, and reasoner, as shown in *Figure 1*. Reasoner is in charge of making local decisions. The device manager and protocol adapter play the role of the SDN control plane and forward plane, which means they control the flow of data to bind into the reasoner and fetch it back to the specific application. The focus of this paper is mainly on the reasoner module in great detail. To develop an entire IoT gateway by leveraging new software technologies and architectural concepts, in very much the same way, we implemented device management and protocol adapter according to the suggestions in [13], [14] . In order to automated arrangement and coordination of complex IoT devices, there is a need for an autonomic gateway, which will enable implicitly autonomic control to some extent by fuzzy logic-based rules for the ability to offer reasoning for different context-aware applications as shown in *Figure 2*.

In this paper, we demonstrate the design and development of an autonomic IoT controller with a context-aware reasoning service as a generic enabler to enable smart home automation independent of any particular home automation solution. The next section provides the design and development process of the reasoning service, which automates smart home applications.

3.3. Context-Aware Reasoning as Generic Enabler for Dynamic Response and Real-Time Needs

A large number of small-data are handled by the IoT gateway[15], in which these small data must immediately transform into actionable information to making technical decisions, as shown in *Figure 1* on intelligence plane (e.g., temperature, humidity, sensor data. And so forth). Thus, making-decision and providing context-based services in order to gain distributed intelligence are attracting attention in the IoT. Reasoning, the most important part of decision-making, is about making conclusions and deducing new facts that do not exist in the knowledge base. The primary role of a reasoner is to reason based on the data fed into it [16].

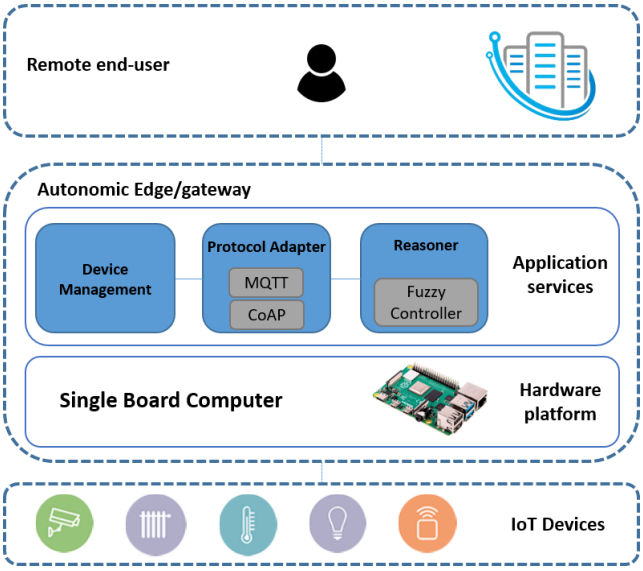

Fig. 2. The Low-Level Intelligent Control Scheme.

To design a reasoning service for context-aware applications that responds to context changes in the network, we designed a Fuzzy Logic Controller (FLC) that responds to context changes. Since FLC, like human logic, has no boundaries and is based on decision making methods, operation control is required for better decision making. The need for operation control has in turn led to the use of an FLC mechanism, that results in *Figure 3*. To make the reasoning engine an optimal predictor of the uncertainty factor of data, which enables the selection of reasoning to reduce the inference time of the sharing process, a generic enabler method is used to map an input to an output by using a logical interval of type 2, as explained in *Section 3.3.3*. In the next section, the fuzzification process model is explained.

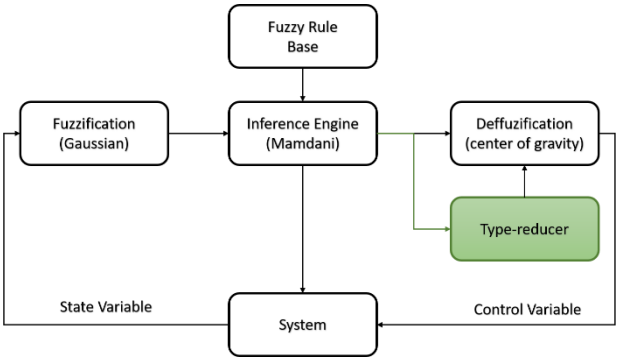

Fig. 3. Fuzzy Interface System.

3.3.1. Fuzzification process for Real-Time Needs

Due to the complexity of the system, parameters in this kind of process model equations are uncertain variables and interact with each other. These parameters include: first, system configuration parameters (i.e., the comfortable temperature values); second, operation conditions (i.e., utilization which is

the quote between the numbers of requests for regulation and reply within a timer event); third, parameters which can only be calculated by interval type-2 which will affect the control cycle time. Since the effect of uncertain parameters is different, some have a significant influence on the system response, and some do not. In order to reduce the computational time, only those parameters which have a significant impact on system behavior, are considered and treated as fuzzy variables in this paper. The data input load to dynamic time-varying is utilized in order to study the speed response of the process model. In this experiment, each universe of discourse is divided, as shown in *Table 1*, *Table 2*, and *Table 3*. Many types of curves can be used (e.g., Gaussian, sigmoid, triangular, trapezoidal, Z-shaped, and so forth), but in this experiment, we used Gaussian (Ga) as membership function. The fuzzy system consists of two input parameters: one, current temperature, *Figure 4(a)* and, two, Estimated Time of Arrival (ETA), *Figure 4(b)*. The former is gathered from the user using a mobile application, and the latter is gathered via the indoor sensor temperature. The system has one output parameter, *Figure 4(c)*, that controls the heater speed of the heating system.

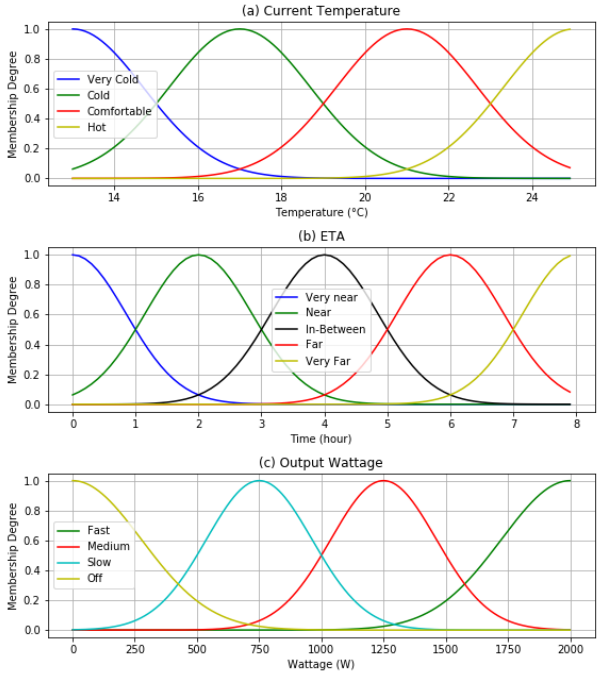

Fig. 4. Type-1 fuzzification and its response to data input (a) Current temp. (b) ETA (c) output.

Table 1. The FLC model universe of discourse.

	MF	Type of MF	Parameters
MF1	Very Cold	Ga	13, 1.69
MF2	Cold	Ga	17, 1.69
MF3	Comfortable	Ga	21, 1.69
MF4	Hot	Ga	25, 1.69

Table 2. The ETA universe of discourse

	MF	Type of MF	Parameters
MF1	Very Near	Ga	0, 0.849
MF2	Near	Ga	2, 0.849
MF3	In-Between	Ga	4, 0.849
MF4	Far	Ga	6, 0.849
MF5	Very Far	Ga	8, 0.849

Table 3. The Speed of Heater universe of discourse

	MF	Type of MF	Parameters
MF1	Off	Ga	0, 276.4
MF2	Slow	Ga	750, 276.4
MF3	Medium	Ga	1250, 212.3
MF4	Fast	Ga	2000, 212.3

Table 4. Normalized effects of uncertain parameters for temperature.

Indoor Temperature / Outdoor Temperature	Cold	Cool	Medium	Hot
Cold	Cold	Cold	Cold	Medium
Cool	Cold	Cool	Cool	Medium
Medium	Cool	Medium	Medium	Hot
Hot	Cool	Medium	Medium	Hot

Table 5. Normalized effects of uncertain parameters for the system.

Temperature / ETA	Very Near	Near	In-Between	Far
Cold	Fast	Fast	Fast	Medium
Cool	Fast	Fast	Fast	Medium
Medium	Fast	Medium	Slow	Slow
Hot	Off	Off	Slow	Slow

3.3.2. Generic Enabler Inference Sharing Engine for Dynamic Response

The Generic Enabler Fuzzy Inference System (GNFIS) is a method of mapping an input to an output using fuzzy logic interval type-2. The GNFIS tries to formalize the context-awareness reasoning process of human language by means of fuzzy logic (by building fuzzy rules).

GNFIS is based on fuzzy inference methods, which are categorized into direct and indirect methods. Direct methods, such as Mamdani's and Sugeno's, are the most commonly used—these two methods only differ in how they obtain the outputs, and indirect methods are more complex [17]. Nevertheless, in this paper, Mamdani's fuzzy inference method was chosen. Mamdani's method is the most commonly used in applications due to its simple structure of min-max operations. It is, therefore, more suitable for the system design of a fuzzy system [17]. The GNFIS controller consists of a plant system as the controlled process with uncertain source parameters, context-awareness changes, and packet forwarding in the gateway. The second part is a type-2 fuzzy controller that has a controlling element which generates control input. The third part is the controlled variable, and the fourth is the feedback control system, which is an output observed information.

3.3.2. Generic Enabler Fuzzy Inference system

With clear indications of context burstiness in edge gateways and to mine the context of such situations, to capture these kinds of context arrival properties, we used dynamic fuzzy modeling to model the time-varying arrival rate and to capture their essential correlation.

This fuzzy approach model allows an uncertainty-based parameter in the mathematic control to incorporate all the uncertainty about their values. These parameters

model and normalize effects of every uncertain parameter, concerning output interests, are computed in two steps. In the first step, the fuzzy rules defined in Table 4 have been applied over two fuzzy values previously obtained, representing the outdoor and indoor temperature. With these fuzzy rules, we obtain the degree of truth of each possible linguistic variable for temperature based on both indoor and outdoor temperature. In the second step, the system can decide about what action to do in order to adjust the room temperature. This is the rationale behind applying another set of fuzzy rules, as depicted in Table 5, where the inputs are the temperature obtained in the fest step and the ETA and the degree of truth of action to do. The benefit of having fuzzy antecedents is to provide a basis for an interpolation mechanism. To accomplish this, the investigation of probability distributions of the simulation results under parameter uncertainties is very important to ensure the accuracy of the model predictions. The interpolation model presented in this paper helps identifying the upper and lower bound model outputs. The next section explains the motivation behind deploying the type-2 fuzzy logic system and its ability to ensure the accuracy of the model predications.

3.3.3. The Components of Type-2 Fuzzy Sets

The basic phenomenon of type-1 fuzzy (T1FLS) arithmetic, which consists of the overestimation of results depending on the actual form of the fuzzy rational

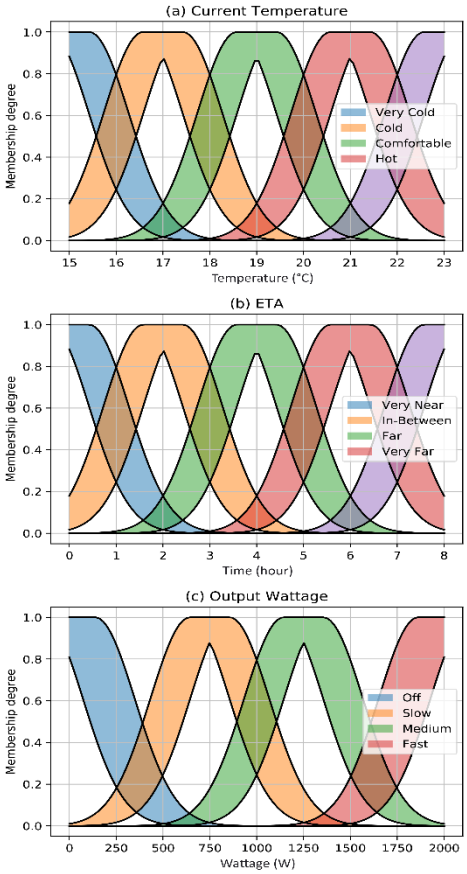

Fig. 5. Type-2 fuzzification and its response to uncertainty data input (a) Current temp. (b) ETA (c) output.

expression, has been shown to be a practical problem [18]. Type-2 FLSs (T2FLSs) is proposed as an extension of T1FLS. While designing a T1FLS, expertise and knowledge are needed to decide both the MFs and fuzzy rules. The T1FLS, whose MFs are type-1 fuzzy sets, is unable to handle rule uncertainties directly. T2FLSs can better deal with the vagueness inherent in linguistic words than T1FLSs. The use of fuzzy MF models the uncertainties. Thus, T2FLSs are a better choice for cases where establish the exact MF for a fuzzy set is difficult, which is very useful to tackle uncertainties [19]. *Figure 5* shows the components of the T2FLS inputs as, one, current temperature, *Figure 5(a)* and, two, Estimated Time of Arrival (ETA), *Figure 5(b)*, and, three, T2FLS output, *Figure 5(c)*. Contrary to type-1 fuzzy in which membership functions in sets are certain, in type-2 fuzzy sets, membership functions are themselves fuzzy.

Therefore, the antecedents and the consequents of the rules in type-2 fuzzy sets are uncertain. While a type-1 membership grade is a crisp number in the interval of [0, 1] as shown in *Figure 3*, a type-2 membership grade can be any subset in the interval of [0, 1] as shown in *Figure 5*, which is called primary membership. besides, in order to define the possibility for primary memberships, there is a secondary membership value for each primary membership value [20]. The secondary membership functions value-range are [0, 1], in generalized T2FLSs, they are uniform functions that only take on the value of 1 in interval T2FLSs. Compared to interval T2FLSs, the computational burden of general T2FLSs is very high.
The interpolation model presented in this paper is deployed by using T2FLS, which is described in *Section 4*, and *Figure 6* depicts the scenario of the application domain. We also investigated the performance and reliability of the approached model to compare it with the T1FLS. The T2FLS shows the ability that it can be applied in an engineering context provided the uncertain parameters are caused by deficiencies in any part of the model due to lack of knowledge. The uncertainties in the models are related to information gaps that can typically be filled by human subjective opinion on the unknown quantities.

3.3.4. GNFIS Resilience Management

The principle autonomic objective of IoT Gateway is to provide a higher recovery level for resilience compared to current IoT Gateways. The Fuzzy resilience controller initially works with the input variables [21], such as the difference between the actual resilience level and the resilience target. The GNFIS has the ability to perform autonomic resilience management based on stability and robust controller. The adaptation and learning mechanism observe the inputs arrive from the control system and adapts the parameters of the controller to maintain resilience management performance even if there are changes in the controller process (i.e., plant) with heterogeneous source data flows, which leads to improve performance over time.

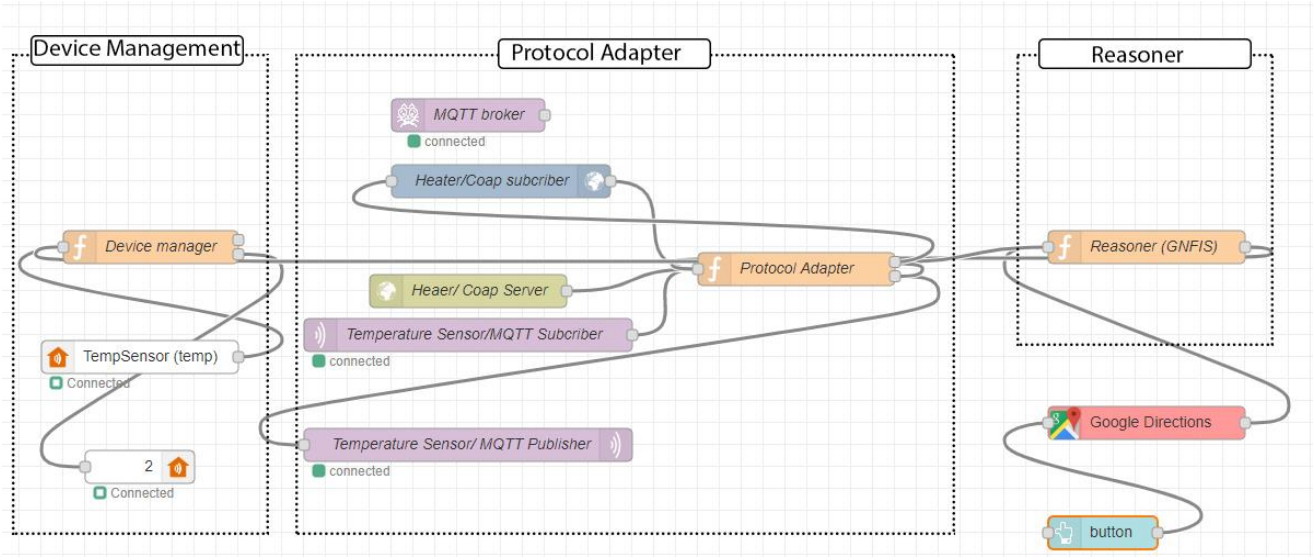

Fig. 6. The node-red based topology scenario.

3.3.5. GNFIS Reliability Management

When designing reasoning for context management, it is important to consider the aspect of reliability. This concern is met with accuracy and synchronization mechanisms in adjusting the output of the GNFIS controller. The controller in the approach is designed so that its learning controller has the ability to improve the performance of the loop controller process and utilizing feedback information from the controller process. The inputs to the controller are fitted with synchronization between different devices and protocols. The controller has the ability to lower the risk of the inputs concerning either the protocols or devices error and to distribute the output. This ability will improve the reliability of the gateway.

3.3.6. Software and Hardware Used in Topology Evaluation

To validate the framework, we defined a topology by using the Node-Red tool [22], which is an open-source flow-based development tool for the integration of IoT hardware devices, APIs (Application Programming Interfaces) and online services developed by IBM Emerging Technology. *Figure 6* shown the node-red flow-based prototype, and it models gateway controller for IoT applications (i.e., smart home reasoner as a flow of information among components). The whole topology is divided into four components, a user interface, a reasoner, a protocol adapter, and device controller/management.

In order to evaluate the controller component behavior, we deployed our gateway in an uncertain data flow environment with Raspberry Pi 4 Single Board Computers (SBC) with 1.5 GHz 64-bit quad-core ARM Cortex-A72 processor and 4GB RAM, running Linux Raspbian Buster with desktop and recommended software. For the device controller, a Tellstick Net [23] is used, and it is a device that lets you remotely control your connected electronics via the Internet. It is compatible with many different remote socket receivers. *Section 4* describes the prototype testing to evaluate effectiveness and responsiveness quantitatively.

4. Performance evaluation

We thoroughly tested the prototype to evaluate the performance of gateway and the reasoning module, along with two primary guidelines:

- Evaluation of output changes of the gateway controller while overloading with data flows to the inputs. In this case, the primary purpose is to evaluate and demonstrate GNFIS effectiveness and resilience quantitatively.
- Evaluation of required bandwidths for the controller compare to the traditional cloud system.
- Evaluation of power consumption of controller compare to the traditional heating system.
- Evaluation of distribution accuracy in the controller process to the output for testing the effectiveness and reliability.

The main goal of this section is to verify and assess the effectiveness of the proposed approach by focusing on a few key aspects of the implemented GNFIS controller. *Figure 7* depicts the scenario of smart home reasoning. The performance results below are based on experimentation over a testbed, which shown in *Figure 6*. The smart home mobile interface suggests the fastest route to cottage location. When a user starts to send a request, the application forwards an activation request to the smart home reasoning in the autonomic gateway. After the confirmation of the activation request, the

gateway starts heating the cottage to module ETA and temperature of the interior. The user is able to check if the heating has started. GNFIS can learn about activation requests and heating gain patterns from historical data and predict for the next activation request to start heating.

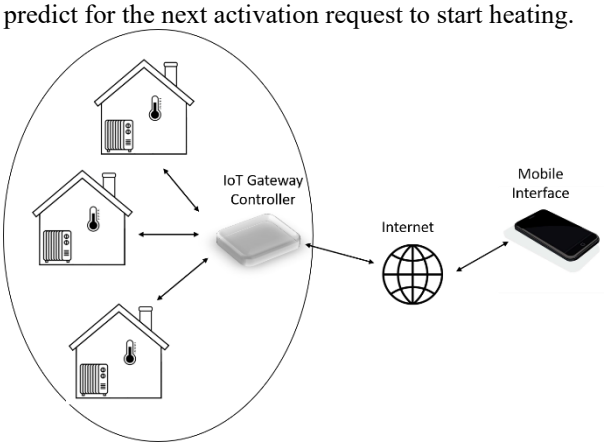

Fig. 7. Illustration of Smart Home Reasoning in the Autonomic IoT Gateway.

4.1. Experiment of Gateway Response time

To evaluate the GNFIS, we developed a Nod-Red based application running on Raspberry Pi 4 (Model B) and Tellstick; as shown in *Figure 6*, it is sending and receiving messages within the activation requested. One of the most noticeable features of edge computing compared to the cloud is latency. Latency shows how fast a gateway has reacted to the devices. For evaluation of the performance GNFIS of the gateway, we measured and compared the latency of reasoning within the activation requested at both cloud and edge (gateway) based on QPS (query per second). As shown in *Figure 8*, the reasoning at the edge has had shorter response times and shorter latency compared to cloud computing. By increasing the QPS from 10 to 40, the response time linearly increases for both edge and cloud, which is 200 ms and 900 ms, respectively. The response time starts to increase exponentially, after 24 QPS, for the cloud to more than 4000 ms at 60 QPS. With considering 1 second as acceptable latency, we can see the gateway can support at most 56 QPS, and the cloud can handle 41 QPS. *Figure 9* shows the same trend as *Figure 8*. When the number of edge devices is increased to 60, the latency quickly increases to about 1200ms. Consequently, 50-56 edge devices are the maximum loads for a gateway. These preliminary results show that providing low latency and low-level of the knowledge to reduce dependency on cloud is possible at the edge (gateway with GNFIS) of the IoT.

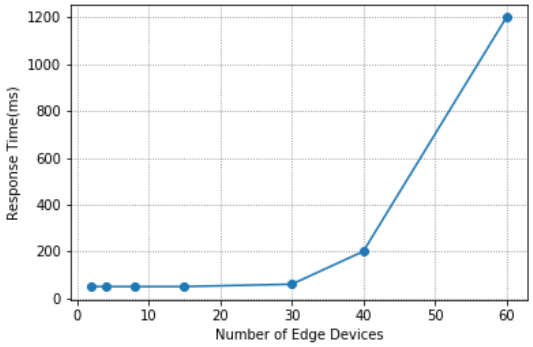

Fig. 8. Edge and cloud reasoning response time.

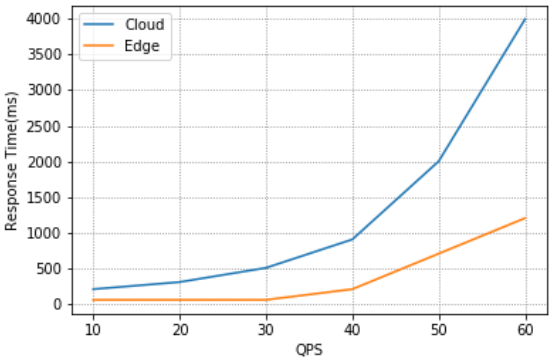

Fig. 9. Scalability in response time.

4.2. GNFIS Required Bandwidth

In order to investigate the required bandwidths, it is assumed that the four bandwidths for uploading service demand responses is $(5, 10, 15, 20) * 104$ bits per second and their transmission distance is different. Among them, the transmission distance of service demand responses cloud center architecture require is much longer than the edge computing architecture. As shown in Figure 10 the edge computing architecture require less that bandwidth compare to the cloud.

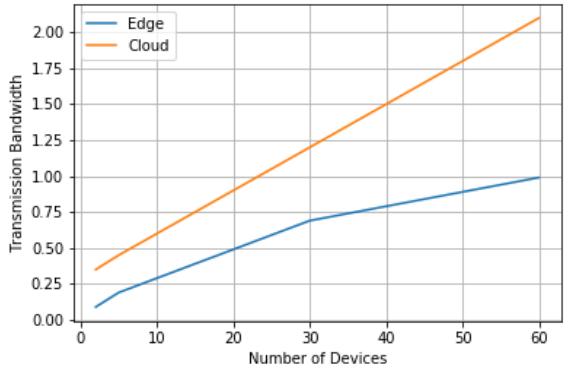

Fig. 10. The required bandwidth.

4.3. GNFIS Electricity Consumption.

In this paper, only the heating electricity consumption is considered to evaluate the electricity consumption. Heater electricity consumption is higher if high voltage is used. As you can see in the *Figure 11*, we

have divided the power of the heater into 4 effective modes (such as the fuzzification part) zero (off), one(slow), two(medium), four(fast).

For simplification in this figure, whenever the effective mode change is included. Our proposed system requires the use of higher hairs compared to older home heating systems.

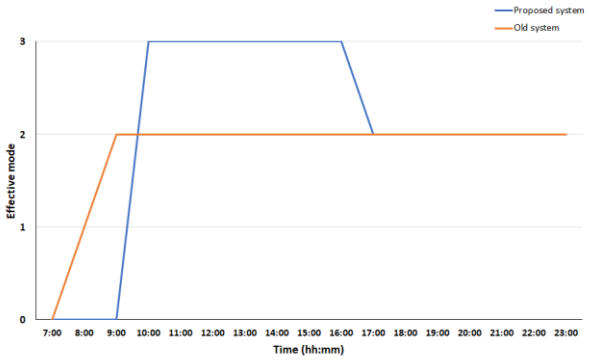

Fig. 11. Effective mode of each system for one day.

4.4. GNFIS Reasoning Performance

In order to evaluate the GNFIS controller, we compared its accuracy with a proportional integral derivative controller (PID) based on the statistical parameters such as the root-mean-square error (RMSE), and Mean absolute percentage error (MAPE) by comparing predicted and measured values of GNFIS. The RMSE and MAPE are defined by the Equation (1) and (2) respectively. *Table 5* shows the result. The GNFIS controller has resulted in lower RMSE and MAPE compared to the PID.

$$\text{RMSE} = \sqrt{\frac{1}{n} + \sum_{i=1}^{n} (y_{pred_i} - y_{obs_i})^2} \quad (1)$$

$$\text{MAPE} = \frac{100\%}{n} + \sum_{i=1}^{n} \frac{\left| y_{pred_i} - y_{obs_i} \right|}{y_{pred_i}} \quad (2)$$

Table 5. Performance characteristics of heating system with Fuzzy Type 1, 2, and PID.

Controller	RMSE	MAPE
PID controller	0.325	5.365
Fuzzy type-1Controller	0.108	4.823
Interval Fuzzy type-2 Controller	1.942	4.064

5. Conclusion

We are experiencing a significant problem of distributed IoT gateways that continuously perform localized analytics and leverage the model learned from large-scale data for IoT intelligence. In this paper, we present an architecture for a distributed intelligent gateway controller in a distributed IoT framework that addresses the challenge of building a distributed intelligent controller for resilient, reliable, and low-latency intelligent control in distributed IoT. The proposed approach is designed for the purpose of building an edge intelligent controller to provide low-level intelligence, and the cloud controller would provide high-level intelligence. A simulated smart home scenario has verified the plausibility of the proposed approach on a Raspberry Pi 4 Model B as a proof-of-concept. Moreover, this paper presented the fuzzy logic system as a reasoner to make decisions. The results show that the use of a fuzzy logic system significantly reduces the latency and increases the accuracy of edge controller. As a complement to the work presented in this paper, an outstanding work that can be done in the future is a distributed edge gateway controller for large-scale IoT applications.

References

[1] A. Botta, W. de Donato, V. Persico, and A. Pescapé, “On the Integration of Cloud Computing and Internet of Things,” in *2014 International Conference on Future Internet of Things and Cloud*, Aug. 2014, pp. 23–30, doi: 10.1109/FiCloud.2014.14. https://doi.org/10.1109/FiCloud.2014.14

[2] A. Antonić, M. Marjanović, K. Pripužić, and I. Podnar Žarko, “A mobile crowd sensing ecosystem enabled by CUPUS: Cloud-based publish/subscribe middleware for the Internet of Things,” *Future Generation Computer Systems*, vol. 56, pp. 607–622, Mar. 2016, doi: 10.1016/j.future.2015.08.005. https://doi.org/10.1016/j.future.2015.08.005

[3] J. Soldatos *et al.*, “OpenIoT: Open Source Internet-of-Things in the Cloud,” *Lecture Notes in Computer Science*, pp. 13–25. https://doi.org/10.1007/978-3-319-16546-2_3

[4] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, “Context Aware Computing for The Internet of Things: A Survey,” *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 414–454, First 2014, doi: 10.1109/SURV.2013.042313.00197. https://doi.org/10.1109/SURV.2013.042313.00197

[5] L. Ibarra and C. Webb, “Advantages of Fuzzy Control While Dealing with Complex/ Unknown Model Dynamics: A Quadcopter Example,” *New Applications of Artificial Intelligence*, Aug. 2016, doi: 10.5772/62530. https://doi.org/10.5772/62530

[6] C. Allen, I. Smit, and W. Wallach, “Artificial Morality: Top-down, Bottom-up, and Hybrid Approaches,” *Ethics Inf Technol*, vol. 7, no. 3, pp. 149–155, Sep. 2005, doi: 10.1007/s10676-006-0004-4. https://doi.org/10.1007/s10676-006-0004-4

[7] A. I. Maarala, X. Su, and J. Riekki, “Semantic Reasoning for Context-Aware Internet of Things Applications,” *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 461–473, Apr. 2017, doi: 10.1109/JIOT.2016.2587060. https://doi.org/10.1109/JIOT.2016.2587060

[8] Z. Shi, *Advanced Artificial Intelligence*. World Scientific, 2011. https://doi.org/10.1142/7547

[9] J. Urbani, S. Kotoulas, E. Oren, and F. van Harmelen, "Scalable Distributed Reasoning Using MapReduce," in *The Semantic Web - ISWC 2009*, Berlin, Heidelberg, 2009, pp. 634–649, doi: 10.1007/978-3-642-04930-9_40. https://doi.org/10.1007/978-3-642-04930-9_40

[10] M. Assel *et al.*, "Large knowledge collider: a service-oriented platform for large-scale semantic reasoning," in *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*, Sogndal, Norway, May 2011, pp. 1–9, doi: 10.1145/1988688.1988737. https://doi.org/10.1145/1988688.1988737

[11] E. Della Valle, I. Celino, D. Dell'Aglio, R. Grothmann, F. Steinke, and V. Tresp, "Semantic Traffic-Aware Routing Using the LarKC Platform," *IEEE Internet Computing*, vol. 15, no. 6, pp. 15–23, Nov. 2011, doi: 10.1109/MIC.2011.107. https://doi.org/10.1109/MIC.2011.107

[12] H. Rahman and R. Rahmani, "Enabling distributed intelligence assisted Future Internet of Things Controller (FITC)," *Applied Computing and Informatics*, vol. 14, no. 1, pp. 73–87, Jan. 2018, doi: 10.1016/j.aci.2017.05.001. https://doi.org/10.1016/j.aci.2017.05.001

[13] R. Firouzi, R. Rahmani, and T. Kanter, "An Autonomic IoT Gateway for Smart Home Using Fuzzy Logic Reasoner," *Procedia Computer Science*, vol. 177, pp. 102–111, Jan. 2020, doi: 10.1016/j.procs.2020.10.017. https://doi.org/10.1016/j.procs.2020.10.017

[14] R. Rahmani and R. Firouzi, "Gateway controller with deep sensing: learning to be autonomic in intelligent internet of things," *International Journal of Communication Networks and Distributed Systems*, vol. 26, no. 1, pp. 1–29, Oct. 2020, doi: 10.1504/IJCNDS.2021.111631. https://doi.org/10.1504/IJCNDS.2021.111631

[15] S. K. Datta, C. Bonnet, and N. Nikaein, "An IoT gateway centric architecture to provide novel M2M services," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, Mar. 2014, pp. 514–519, doi: 10.1109/WF-IoT.2014.6803221. https://doi.org/10.1109/WF-IoT.2014.6803221

[16] H. Rahman, R. Rahmani, and T. Kanter, "Multi-Modal Context-Aware reasoNer (CAN) at the Edge of IoT," *Procedia Computer Science*, vol. 109, pp. 335–342, 2017, doi: 10.1016/j.procs.2017.05.360. https://doi.org/10.1016/j.procs.2017.05.360

[17] E. H. Mamdani and S. Assilian, "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller," in *Readings in Fuzzy Sets for Intelligent Systems*, D. Dubois, H. Prade, and R. R. Yager, Eds. Morgan Kaufmann, 1993, pp. 283–289. https://doi.org/10.1016/B978-1-4832-1450-4.50032-8

[18] M. Hanss, "The transformation method for the simulation and analysis of systems with uncertain parameters," *Fuzzy Sets and Systems*, vol. 130, no. 3, pp. 277–289, Sep. 2002, doi: 10.1016/S0165-0114(02)00045-3. https://doi.org/10.1016/S0165-0114(02)00045-3

[19] M. A. Khanesar, M. Teshnehlab, E. Kayacan, and O. Kaynak, "A novel type-2 fuzzy membership function: application to the prediction of noisy data," in *2010 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, Sep. 2010, pp. 128–133, doi: 10.1109/CIMSA.2010.5611774. https://doi.org/10.1109/CIMSA.2010.5611774

[20] J. M. Mendel, *Uncertain Rule-Based Fuzzy Systems: Introduction and New Directions, 2nd Edition*, 2nd ed. Springer International Publishing, 2017. https://doi.org/10.1007/978-3-319-51370-6

[21] J. R. G. Cárdenas, À. Nebot, F. Mugica, and A. Vellido, "A decision making support tool: The resilience management fuzzy controller," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, Jul. 2016, pp. 2313–2320, doi: 10.1109/CEC.2016.7744074. https://doi.org/10.1109/CEC.2016.7744074

[22] "Node-RED." https://nodered.org/ (accessed Mar. 05, 2020).

[23] "Z-Wave Gateway TellStick ZNet Lite V2 | Telldus." https://telldus.com/produkt/z-wave-gateway-tellstick-znet-lite-ver-2/ (accessed Feb. 10, 2020).