ATLANTIS PRESS International Journal of Computational Intelligence Systems, Vol. 11 (2018) 316–329

Using fuzzy rules for network behavior identification: application for differentiated services in an Ethernet network

Vincent Bombardier [1,2] , Jean-Philippe Georges [1,2] , Éric Rondeau [1,2] , Idriss Diouri [1,2]

[1] Université de Lorraine, CRAN, UMR 7039,
Campus Sciences, BP 70239, Vandœuvre-lès-Nancy Cedex, 54506, France
E-mail: {firstname.name}@univ-lorraine.fr
[2] CNRS, CRAN, UMR 7039, France

Received 28 April 2017

Accepted 4 August 2017

Abstract

The Quality of Service (QoS) offered by a network is based on criteria such as delay, jitter and message loss. QoS management is crucial to ensure that the network respects the communication constraints defined by the application requirements. The network must dynamically manage QoS according to the evolution of network usage. To control QoS, the network implements mechanisms to schedule and regulate traffic. The main issue is identifying the relationship between these mechanisms and their effects on QoS values. The objective of this paper is to propose a method, based on fuzzy logic, for identifying this relationship and reusing it to dynamically control QoS. This study only focuses on the impact of the Weighted Round Robin (WRR) scheduler on the delay in a switched Ethernet network. The fuzzy controller is implemented in a network simulation tool (Riverbed Modeler) to efficiently manage the delay in the network.

Keywords: Fuzzy rules, fuzzy identification, fuzzy control, switched Ethernet network, quality of service, soft scheduling

1. Introduction

Network communications are complex systems that are composed of many interconnected and distributed devices such as routers, switches, firewalls and terminals. These devices implement many services and protocols that are structured according to the TCP/IP or UDP/IP protocol stacks. One of the main challenges in managing a network is the ability to measure and control its Quality of Service (QoS) with respect to the requirements of the application using the network. For measurement, QoS gathers the network performance, which can be expressed in terms of message delay, message jitter, message loss and security, among other factors. To control QoS, networks integrate many buffering mechanisms such as traffic scheduling and traffic smoothing. It is difficult to establish a formal or pseudo-formal relationship between QoS control mechanisms and their real impact on the QoS values. In practice, the tuning of QoS controllers is achieved in a manual and empirical way. In research domains, the correlation between QoS controllers and QoS values is assessed using different mathematical approaches. The queuing theory estimates the mean delay and the network calculus is applied to determine the bounded delay[19]. Petri nets and Specification Description Language (SDL) are other mathematical representations for obtaining quantitative network information. The QoS estimation in these research approaches

Copyright © 2018, the Authors. Published by Atlantis Press.
This is an open access article under the CC BY-NC license (http://creativecommons.org/licenses/by-nc/4.0/).

is based on constructive modeling. The scheduler, the smoother and the network architecture are modeled from elementary components (queue, server and curve) or graphs (states and events). The main limitation of the constructive modeling approach is that it always results in a simplification of the network representation due to the difficulty of representing the entire network complexity in one model.

The objective of this paper is to propose another approach for estimating QoS values during the investigation and identification phases of modeling the network. The network is analyzed as a black box, where the inputs are both QoS controllers and incoming traffic and the outputs are QoS values. The main interest of this approach is to consider, de facto, all of the mechanisms implemented in the network in the modeling step. However, since the identification phase was achieved for a specific network technology, the results of the study will be dependent on the concerned controller.

This paper also analyzes a switched Ethernet network, a technology that is widely used both in administrative and production services. Switched Ethernet also supports the classification of service introduced by the IEEE 802.1p group that enables the differentiation of service messages according to their priority. The general principle of this standard is to associate as many output buffers to each output port of the switch as there are levels of priorities to be managed (in practice, the maximum number of buffers is four). Thus, when an Ethernet switch receives a message, it analyzes both its destination address and its priority, as shown in Fig. 1.

The destination address is used to switch the message to the correct output port, while the priority is used to select the output buffer to store the message. On each output port, a scheduler is implemented, which enables the provision of more or less bandwidth to each buffer. In the first switched Ethernet standards (IEEE 802.1D/Q), only two types of schedulers were defined: the strict priority and the Weighted Round Robin (WRR)[9]. The aim of WRR is to avoid the starvation situation which could appear with strict priority. The WRR mechanism is simple to understand: it associates a weight w_i with each buffer i and processes the buffers

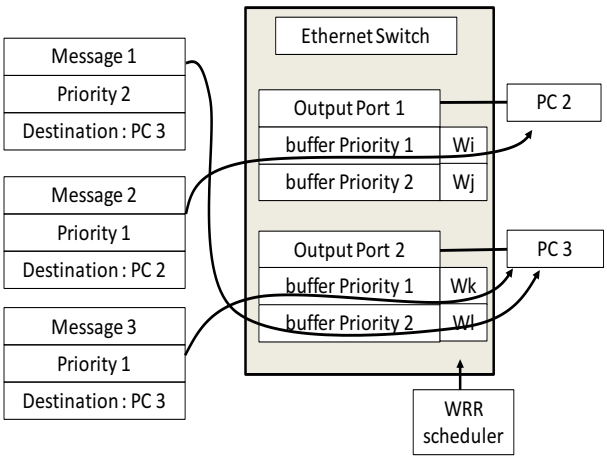


Fig. 1. Ethernet class of services.

in round robin format by forwarding the w_i messages stored in the buffer i. If the buffer is empty, the scheduler automatically processes the next buffer. Thus, when w_i is high, the bandwidth is also high.

Last extensions of the IEEE 802.1 introduce new scheduling algorithms. The specification of the IEEE 802.1 Audio/Video Bridging (AVB) standard[22] includes an alternative to fair queueing like WRR based on a Credit-Based Fair Queuing (CBFQ)[2] traffic shaping and stream reservation. The credit-based shaper operates on one or many outgoing queues per port in the bridge. This algorithm defines credits for two Stream Reservation (SR) classes (Class-A, Class-B). Basically, credits are accumulated when a class waits for service (at a rate of idleSlope) and are spent when it is served (at a rate of sendSlope). Accumulated credits are limited by two parameters (hiCredit and loCredit)[30]. AVB combines also CBFQ for SR classes and strict priority for best-effort traffic[24] and might support a frame preemption pattern (802.1Qbu project).

For AVB, IEEE 802.1Qat[23] defines also a Stream Reservation Protocol that allows applications to dynamically reserve bandwidth in the network and hence guarantees a certain latency for reserved streams. The rate of credit accumulation and release can be hence adjusted on a queue-by-queue basis to produce a weighted queuing behavior. For WRR scheduling in Ethernet switches, no similar mechanism was planned even if Demers details a solution to dynamically tune WRR weights with re-

gards to the traffic evolution.[9] If the tuning of these weights impacts both the QoS performance and the delay, it is however difficult to understand the correlation between the weight and the delay. To avoid this difficulty, we propose to use an approach based on fuzzy set theory, particularly fuzzy logic. This methodology is applied in this paper only for WRR scheduling that is implemented in many legacy (non AVB compliant) Ethernet switches although it is not the most efficient and suffer from a lack of weights setup. Contrary to traffic shaping mechanisms based on fixed interval times, WRR should be also able to maintain at least partially real-time traffic bursts.

Current projects look at improving time guarantees, in particular the IEEEE P802.1Qbv that introduces the Time Aware Shaper. It blocks the non schedule traffic, so that the port is idle when the schedule traffic is scheduled for transmission. This corresponds to a basic form of time-triggered communication as it is defined by Time-Triggered Ethernet switch.[44] The switch supports time-triggered (according to SAE AS 6802), rate-constrained (ARINC 664, part 7), and COTS Ethernet (IEEE 802.3) traffic flows. Time triggered domain needs to be shared among all TTEthernet participants and switches. Allocations are here defined on switches in traffic config tables. Even such switches are compatibles with 802.1D/Q/AVB switches, time-triggered might requires specific interfaces.

Fuzzy set theory was selected for three reasons. (1) A rapid preliminary study on the relationship between the delay and weight assigned to a WRR scheduler showed non-linearities. (2) The user requirements can be expressed by non-experts in network. For example, in an IP phone context, the user can define the quality of a conversation (e.g., good, correct, nosy, bad) but may lack the competence to translate the quality of the conversation in terms of message delay. The assessment of an IP phone conversation also depends on the group of interviewed users reaching uncertain results in the translation. (3) The fuzzy set theory both enables the identification of the network behavior and reuses this identification to control the QoS tuning.

Some studies on networks using fuzzy set theory[53] and fuzzy logic have been published. These studies present models that describe the relationship between the users and the communication network, such as the Quality of Experiment (QoE) model[20] and QoS provisioning, as regards Service Level Agreement (SLA) specification[13,14,10]. The fuzzy logic methods and tools have also been applied to QoS management for wireless sensors[49,52,17] and ad-hoc mobile networks[28,43]. Other researchers have developed strategies based on fuzzy logic for congestion control,[5,41,47] traffic smoothing,[31] scheduling policies,[39,45] routing protocols,[34,50] network energy consumption[15] or data distribution in Internet of things[29]. Generally, these studies justify the selection of fuzzy logic because it facilitates an overall grasp of the network complexity. These studies are primarily based on the fuzzy controller concept developed by Mamdani[32] and compensate for the imprecision and uncertainties of the communication system. In most studies on this subject, the fuzzy rules are empirically determined.

The originality of the research presented in this paper in comparison to these previous works is its determination of the fuzzy rules of the controller based on automatic modeling rather a human expertise and its identification of network behavior using series of numerical simulations generated using the Riverbed Modeler software, which is a network simulation tool. The latter approach relies on a classification method in which fuzzy linguistic rules are automatically generated. The identification process is then assessed according to a confidence index to only select the relevant rules in the controller.

The organization of this paper is as follows: Section 2 describes both the general methodology used in this work and one application context. Section 3 explains the identification phase and provides the relationship between the delay and weight tuning in the WRR scheduler. Based on the results obtained in the identification phase, Section 4 develops a network fuzzy controller. This controller is implemented in the Riverbed Modeler tool and provides observations of the impact of the WRR scheduler on network delay. The simulation results are provided in section 5. The paper is then concluded with discussions and some perspectives.

2. Methodology

Because the overall objective of this paper is to control the network QoS using an online automatic configuration of different network parameters, we propose the methodology shown in Fig. 2.

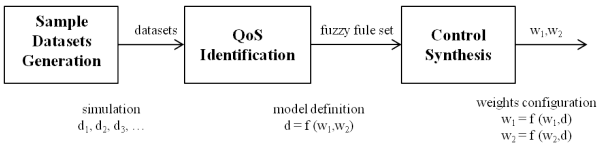


Fig. 2. Methodology.

The first step consists of generating sample datasets that will be used to identify the service offered by the network without referring to human expertise. Thankfully, the datasets may be generated by experimentation or simulation. Simulation has been selected here because it enables the coverage of a larger panel of analysis. Section 3.1 details the generation of switch crossing delay (the QoS indicator of interest here), sample datasets (d_1, d_2, d_3, ...) in relation to the WRR weights (the controllable parameters), and other network parameters (e.g., the traffic load or throughput).

The sample datasets are then analyzed to define a model of the QoS offered by the network. The QoS identification is divided into two parts. As shown in section 3.3, linguistic variables are first defined according to the application requirements in terms of acceptable and non-acceptable QoS. Section 3.4 highlights how the linguistic rules and the datasets are then considered together to generate the fuzzy rule set. A rule associates a delay characteristic to a configuration level of the weights w_1 and w_2.

Generally, the fuzzy identification is then used to build an observer. However, the last step consists of synthesizing a controller based on the fuzzy rule set. It leads to a network fuzzy controller that will select the values to apply locally on a given network component to assess the QoS service level that is expected on the device. In the case study framework of this paper, this means determining which values for weights w_1 and w_2 will provide the expected crossing delay d, as presented in section 4.

Consider the application context that will be used as a case study to illustrate the proposed methodology. It corresponds to a data flow being transported between two end devices on the specific network shown in Fig. 3.

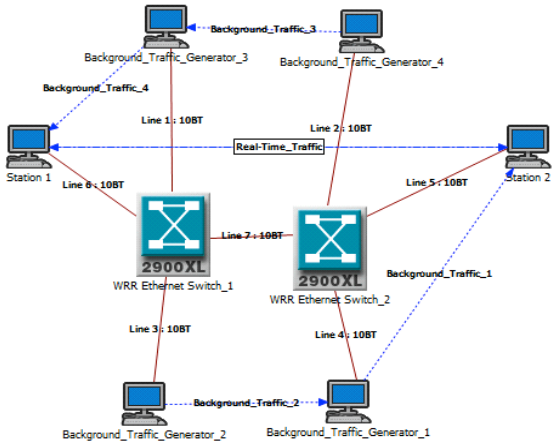


Fig. 3. Application context and network of interest.

To interconnect the end devices of the application (stations 1 and 2) and support the relevant traffic (called real-time traffic), two Ethernet switches are fixed. These two switches support the classification of service mechanisms and the weighted round-robin policy. As a case study of the proposed methodology and since we will consider a small real-time traffic (about 100 kb/s), the initial throughput of the links is limited to 10 Mb/s whereas Ethernet bandwidth may reach 100 Mb/s or 1 Gb/s. For such higher rates, the methodology would obviously been applied for larger traffic loads. The network architecture is often shared by other applications that will compete with the application of interest. Four supplementary stations called Background_Traffic_Generator will thus add four background traffics that will consequently reduce the QoS that the network devices (the two switches) offer for the real-time traffic.

The real-time application requires that the delay it faces remain limited regardless of the load added by the traffic generators. The objective for the network administrator is therefore not only to tune the WRR weights on the two switches such that the delay supported by the real-time traffic is acceptable but also to avoid a starvation situation in which very

little (or even no) bandwidth is offered to the traffic generator. Because the background traffic varies, this optimization should be performed online.

Stations 1 and 2 exchange messages containing 125 bytes every 10 *ms*. The messages are tagged with a high class of service. Background traffic loads the network with a low class of service tag. These traffics correspond to frames of 1500 bytes that are sent using an exponential law, resulting in a mean load equal to 7 Mb/s. In the context of this scenario, the real-time application requires that the high-class message delays must be less than 10 *ms*. This time is divided into four parts: the delay required to cross switch 1, the delay required to cross switch 2, the delay impacted by the source station (e.g., the operating system, TCP/IP or UDP/IP protocol stack) and the delay impacted by the destination station. The delays induced by both stations have been estimated using the Riverbed Modeler tool at 1.44 *ms*. Thus, the bound time to cross the two switches is 8.56 *ms*. The link propagation time is assumed to be negligible.

The bound delay to cross one switch is 4.28 *ms* and values exceeding this limit are unacceptable. The automatic control of the weights that are defined in this paper must reduce the delay as much as possible to prevent the delay from reaching the limit of 4.28 *ms* for one switch and, consequently, an end-to-end delay of 10 *ms*. In this paper, we target to have in a preferred way an end-to-end delay that is one-quarter the size (i.e., 2.5 *ms*). This assumption indicates that the local delay for crossing one switch for high-class frames might be lower than 0.53 *ms* ($0.53 \times 2 + 1.44 = 2.5$ *ms*). Under this second threshold, real-time traffic is served in the preferred way, but the background traffic (lowest class) may face weak network availability. When it has a value between the two thresholds (i.e., 2.5 and 10 *ms*), the QoS is better for the background applications but remains sufficient for the real-time traffic.

This case study will be used to illustrate the methodology proposed in Fig. 2. The next section addresses the first step, i.e., the generation of delay sample datasets relative to the w_1 and w_2 scheduler weights. The values will be obtained through simulations using the Riverbed Modeler tool. Because control of the WRR weights will be achieved on each switch, the network architecture shown in Fig. 3 is simplified to one single switch and two traffic flows, as shown in Fig. 4.

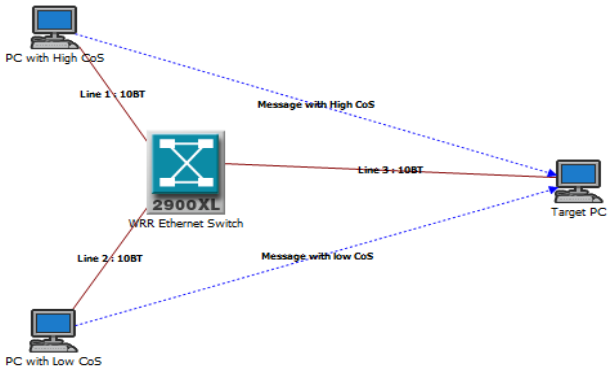


Fig. 4. Simulation system used to identify WRR Ethernet switch behavior.

3. QoS identification using fuzzy rules

Due to the complexity and multiplicity of the mechanisms used in the network, the knowledge of the communication system is rather incomplete. The QoS criterion used in this study is the delay. The objective of network delay identification is to be able to determine the relationship between the delay for crossing a switch and the weights of the WRR scheduler. Thus, the goal is to provide an interpretable model that can be used in the network controller step.

3.1. Data set generation

For each WRR weight configuration and considering the Riverbed Modeler tool recommendations, 30 runs with different seeds have been simulated. In addition, 314 weight configurations have been tested for different (background) traffic loads (5, 6, 7 Mb/s). Each simulation requires five minutes to complete an analysis of the end-to-end delay in steady state. The identification of the delay is based on the mean delays with a confidence index of 90%. The results are shown in Fig. 5 (5 Mb/s), 6 (6 Mb/s) and 7 (7 Mb/s); these figures show the end-to-end delays of the traffic with a high service class based

on the tuning of w_1 and w_2. The weight w_1 corresponds to the weight of the high service class and weight w_2 corresponds to the low service class. The results highlight the impact of the weight tuning on the communication's temporal behavior and show non-linearities that justify the use of fuzzy logic in the identification phase.

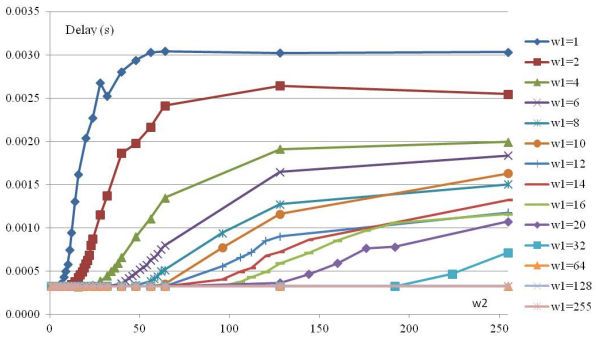


Fig. 5. Switch delays function to w_1 and w_2 with a traffic load of 5 Mb/s.

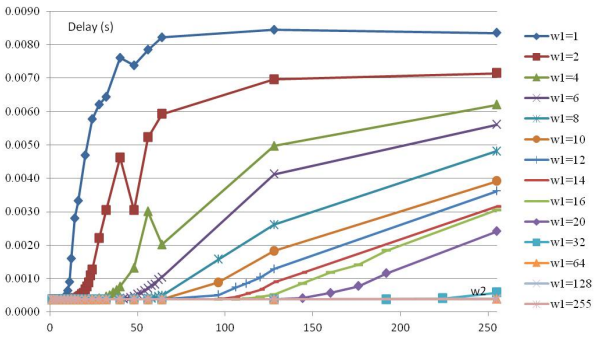


Fig. 6. Switch delays function to w_1 and w_2 with a traffic load of 6 Mb/s.

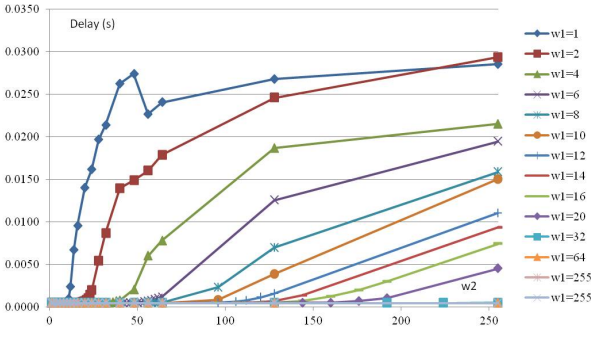


Fig. 7. Switch delays function to w_1 and w_2 with a traffic load of 7 Mb/s.

3.2. *Identification method*

The main idea is first to apply pattern recognition or the classification methods that are used in the image processing domain to the network domain. Then, the weights of the WRR scheduler are considered primitive inputs, and the delay is considered an output of the classification method.

The required interpretability does not permit the use of black box methods such as Neuronal Networks (NNs)[48] or Support Vector Machines (SVMs)[21]. We choose to use a rule-based system,[40,35,36] which seems to be more appropriate in the network context. The obtained rule set will provide knowledge about the delay behavior, with each rule set describing the perceived delay relative to the system tuning.

We use the deductive reasoning mechanism because the system outputs depend on the system inputs. Such rules can be classified into two categories: conjunctive rules and implicative rules[26]. The former contain the possibility and anti-gradual rules, and the latter contain the certitude and gradual rules[11]. The conjunctive rules are derived from the data analysis field, in which the reasoning mechanisms are led by data, whereas the implicative rules are primarily utilized in the cognitive sciences, where reasoning is led by knowledge[12]. The approach utilizing implicative rules is not adopted in this study. Thus, the analysis of the simulation data for delay identification in this paper is based on conjunctive possibility rules[26]. Moreover, this type of rule ensures rule set coherency.

There are many methods that obtain their rule sets from data, including Decision Tree Methods (DTMs)[33] and Genetic Algorithms[7]. Due to the small set of data that can be obtained because of the excessive computing time, the fuzzy approach is chosen. The major issue with these methods is that they require a large amount of simulation data. Thus, we choose the Fuzzy Rule Classifier (FRC) method,[42] which is well adapted for modeling a system with a reduced dataset. The performance of this approach in terms of generalization has been compared to other classifiers (K-NN, NN, SVM, GA, and DTM)[4].

The Fuzzy Rule Classifier (FRC) is based on a supervised learning mechanism that is composed of two steps: the training step to obtain the system model and the generalization step to classify the unknown data.

To identify the rules of the Network relation between the scheduler weights and the delay, we use the training step of the FRC method. This step is divided into two levels: the fuzzy linguistic variable definition (fuzzification) and the fuzzy rule generation, which requires an iterative step for model adjustment.

3.3. Fuzzy linguistic variable definition

The WRR scheduler weights w_1 and w_2 correspond to the input of the system to be identified. The goal of this step is to translate these numerical inputs into linguistic variables[55]. In this case, the tuple (V,X,Tv) represents the following:

- V is the variable (w_1,w_2) defined in the referential X, called the Universe of Discourse.
- The WRR weights are between 1 and 255. The variation domain is then $X = [1,255]$.
- Tv is the vocabulary selected to define the value V linguistically.

The finite or infinite set $Tv = \{A_1,A_2\ldots\}$ contains the normalized fuzzy sub-sets of X used to characterize the variable V. Each A_i is defined by a degree of belonging $\mu A_i(x)$. In this study, the delay is described using three terms: Target Delay (TD), Intermediate Delay (ID) and Unacceptable Delay (UD) (see Fig. 10). Thus, these variables are expressed by a vector that is composed of three membership degrees: $[\mu TD(x),\mu ID(x),\mu UD(x)]$.
In the fuzzification step, the number of variable terms, their distribution in the universe of discourse and their shape must be defined.

The choice of membership function shape has an important role in the fuzzification step and can affect the recognition rate. Three basic shapes of membership function, i.e., triangular, trapezoidal and Gaussian, can be combined. In this study, the Trapezoid-Triangular membership functions are used because they generally give better results[4].

The other difficulty lies in choosing the position of the membership function that defines the linguistic terms. An easier way to choose is to use a regular distribution of the terms in the universe of discourse, but this may provide more terms than are needed, thereby increasing the number of generated rules and the complexity of the model. Another way is to use various methods to achieve an automatic adaptation of Fuzzification. These methods are generally based on genetic algorithms[6,27] or clustering algorithms[8]. However, these techniques require a large amount of data during the learning process. Schmitt et al. proposed an automatic fuzzification method that can be used in the case of reduced sample sets. This method is based on the study of output class typicalities[16]. However, it generates a great number of terms and is therefore inappropriate in the context of this study.

The method proposed in this paper imposes a number of terms equal to the number of output classes, and the locations of the kernels are calculated using the input data belonging to the main class of their terms. The slope and position of term support are defined with a cross for a membership degree equal to 0.5. The main interest of this approach is to provide a high recognition rate with a minimal number of terms and then to improve the interpretability of the obtained rule set. Fig. 8 and 9 show the fuzzification of w_1 and w_2, respectively, using this approach.

The output of the system must also be fuzzified. To define the output linguistic variable, we use the time constraints of the system given in section 2. Based on the definitions of Unacceptable and Target delay, we create the term Intermediate delay. The threshold between Unacceptable and Intermediate delay is set to 4.28 *ms*. In the same way, the initial limit between Target delay and Intermediate delay (Fig. 10) is fixed at 0.53 *ms*. As detailed in the section 2, these both thresholds correspond to the application constraints determined from network simulation tool (Riverbed Modeler).

However, the second threshold limit (0.53 *ms*) has been empirically changed to improve the recognition rate. Based on an analysis of the datasets, this value is set to 0.458 *ms* and respects the application

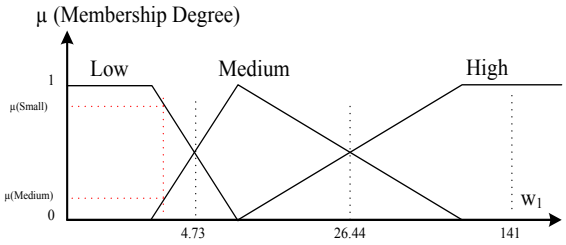


Fig. 8. Three-term fuzzification of the w_1 weight.

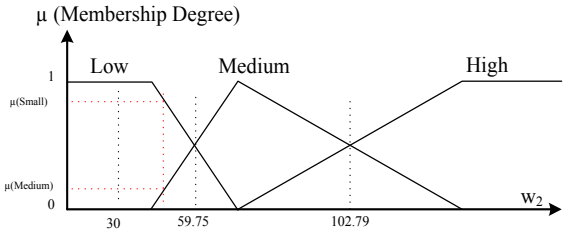


Fig. 9. Three-term fuzzification of the w_2 weight.

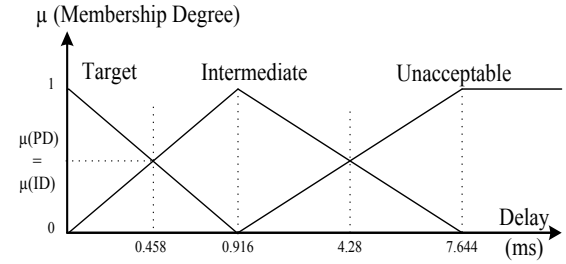


Fig. 10. Fuzzification adjustment of the delay.

constraint (less than 0.53 *ms*). The kernel position of the intermediate term (0.916 *ms*) is automatically obtained through geometric construction. It corresponds to the intersection of the membership functions equal to 0.5, as shown in Fig. 10.

3.4. Fuzzy rules set generation

The second step of the study is to obtain a rule set from the datasets. Several methods can be used to automatically generate the linguistic rules, including Genetic algorithms,[6,1] Wang and Mendel's algorithm,[51] which is used in the network domain[14] and Ishibuchi's algorithm [25].

The Fuzzy Rule Classifier (FRC) that we use here is based on Ishibushi's Algorithm. This method uses an inference mechanism based on Larsen, which is better than the Mamdani inference, especially when there are several premises[3]. The final output is obtained by privileging the rule giving the highest response. Each rule is activated in parallel, and a disjunction operator is used to aggregate the partial conclusions and determine the final decision. The disjunction operator selected is the maximum.

The previously obtained rule set is then adjusted by associating a Confident Factor (CF) with each rule. The adjustment step corresponds to the iterative part of the algorithm[37]. If the classification rate is inferior to a predefined threshold, the iterative procedure tunes the model. If the current sample does not match the rule, the CF coefficient decreases. Otherwise, if the sample confirms the rule, CF is increased.

Table 1 summarizes the recognition rates obtained from the rule set after the teaching step. The global rate is 80.36%, which shows good model behavior. The small number of samples obtained for some delay classes explains the poor scores obtained for the intermediate and unacceptable delay classes. Finally, the model is interesting because it shows a behavior that is independent of the network load.

Table 1. Recognition rates

Data Set	5 *M*	6 *M*	7 *M*	Total
TD	228	233	231	692
	98.70%	97.40%	97.00%	97.70%
ID	86	62	50	198
	40.70%	45.20%	38.00%	41.40%
UD	0	19	33	52
		31.60%	18.20%	23.10%
Total	82.80%	83.12%	79.30%	81.74%

The rule set is given in Table 2 and is issued from the iterative adjustment step. One rule can be considered unambiguous if its CF coefficient is close to 1. Conversely, the rules associated with a low CF correspond to an area where the model is ambiguous.

Table 2. Generated rule set

Rule Number	w_1	w_2	Delay	CF
0	L	L	TD	0.125397
1	M	L	TD	0.676380
2	H	L	TD	0.999970
3	L	M	ID	0.135974
4	M	M	TD	0.089385
5	H	M	TD	0.808859
6	L	H	UD	0.289608
7	M	H	ID	0.352407
8	H	H	TD	0.643659

Thus, rule number 2, which can be expressed as if the weight w_1 is high and the weight w_2 is small, then the delay complies with the target delay value, has a pertinent confidence index because its score is equal to 0.99997. Conversely, rule number 3 is on the Intermediate delay and is not pertinent because it has a CF equal to 0.135974. Thus, rule number 3 is discarded and only the rules with a high CF are selected for tuning the WRR scheduler.

4. Fuzzy controller step

The goal of this phase is to propose a controller capable of maintaining the end-to-end delay for the high-class traffic that is acceptable according to the application requirements. At the same time, the controller has to maximize the bandwidth for other traffic. Based on the fuzzy rule modeling, we chose to implement these rules in a fuzzy controller.

Fuzzy logic control was originally introduced by Zadeh[54] and was developed as a model-free control design approach. It has since had great success in industry applications in various domains,[38] including engine speed regulation, autofocus control and washing machine regulation. However, it is not usually applied in the network domain.

Over the past ten years, prevailing research efforts on fuzzy logic control have been devoted to model-based fuzzy control systems that guarantee not only the stability but also the performance of closed-loop fuzzy control systems[18]. Fuzzy controllers are particularly applicable when the identification of the control system is too complex to model using classical mathematical tools or when the knowledge of the system cannot be expressed in a numerical way[54]. The main advantages of Fuzzy Control are as follows[46]:

- its simplicity, flexibility and ease of adaptation regarding the system operational conditions. Generally, few rules are sufficient to describe the system.
- the ease of synthesizing the views of several experts.
- the possibility of managing several objectives.
- the robustness against disturbance.

However, the fuzzy controller is not generic and can be developed for specific use cases. For example, the results presented in this paper are specific to the network technology and the application constraints.

The strategy for reconfiguring the weights can be expressed in the following way.
When the high class traffic delay is 100% unacceptable, i.e., it is greater than 7.644 *ms*, rule 2 is applied to retrieve the target delay value. Rule 2 consists of a High w_1 and a Low w_2. Fig. 8 and 9 show that the middle of the high area for w_1 is 141 and that the middle of the low area for w_2 is 30.
When the delay is less than or equal to 7.644 *ms*, the reconfiguration strategy relies on the fuzzification of w_1 and w_2, as given in Fig. 11.

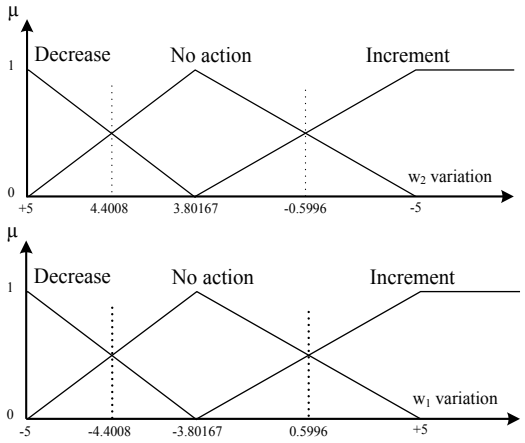


Fig. 11. Fuzzification of w_1 and w_2 weights for a delay less than or equal to 7.644 *ms*.

The control consists of incrementing or decrementing the weights with a maximum step of 5 (or −5). The weights of WRR are integers. Thus, the weight step variation is estimated using the following equations.

If $(UD) > 0$,

$$w_1 = w_1 + \lceil -3.80167 + \mu(UD) \times (5 + 3.80167) \rceil$$
$$w_2 = w_2 + \lfloor 3.80167 + \mu(UD) \times (-5 - 3.80167) \rfloor$$

else if $(UD) = 0$,

$$w_1 = w_1 + \lceil -3.80167 + \mu(UD) \times (-5 + 3.80167) \rceil$$
$$w_2 = w_2 + \lfloor 3.80167 + \mu(UD) \times (5 - 3.80167) \rfloor$$

For example, if the delay is equal to 5 *ms* ($\mu(UD) = 0.5$), the equation becomes:

$$w_1 = w_1 + \lceil -3.80167 + \mu(UD) \times (5 + 3.80167) \rceil$$
$$= w_1 + \lceil 0.599 \rceil = w_1 + 1$$

and

$$w_2 = w_2 + \lfloor 3.80167 + \mu(UD) \times (-5 - 3.80167) \rfloor$$
$$= w_2 + \lfloor -0.599 \rfloor = w_2 - 1$$

5. Simulation results

5.1. *Fuzzy controller in Riverbed Modeler*

The fuzzy controller is implemented in the simulation Riverbed Modeler tool for its evaluation. The simulation Riverbed Modeler tool is structured with three representation levels that are used to compute the network device behavior. In simple terms, the first level defines the process, the devices' input/output ports and their relationships. The second level uses the state chart model to describe the process. Finally, the third level presents associating behaviors written in C++ code describing the states and transitions of the state chart model. The fuzzy controller was tested for the Cisco 2924XL switch model contained in the Riverbed Modeler model library (Fig. 12). At level 1 of this model, a specific link (statistic wire) has been added, which enables the time required to cross the switch to be obtained. The fuzzy controller is located at level 2 of the Cisco 2924XL switch model and is written in C++ code.

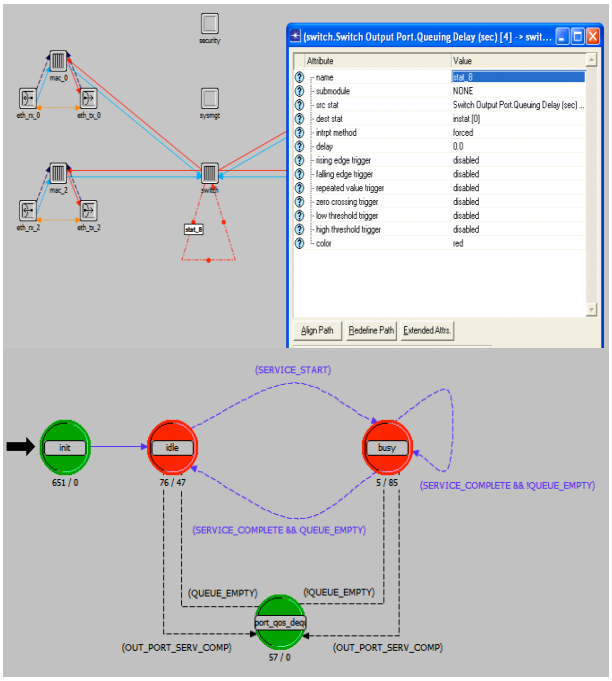
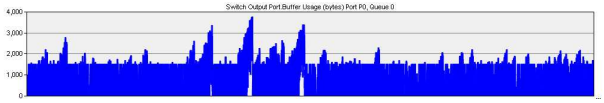


Fig. 12. Fuzzy controller implemented in Riverbed Modeler.

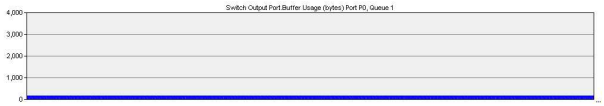
5.2. *Behavior analysis of the fuzzy controller*

The application scenario presented in Fig. 3 is modeled using the Riverbed Modeler simulator. The background traffic loads the network and has a low class of service (dotted arrows in Fig. 3). The background traffic here corresponds to the frames that are periodically sent using an exponential law, resulting in a mean load equal to 7 Mb/s. The fuzzy controller is configured to ensure that the delay of high class frames does not exceed 4.28 *ms*.

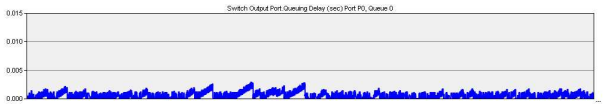
Two simulations have been run. The first simulation shows the network behavior with fixed weights (Fig. 13), and the second simulation shows the network behavior with weights that are dynamically tuned with the fuzzy controller (Fig. 14). It is important to note here that the first scenario assumes a high expert knowledge level because the weights are conservatively defined a priori.



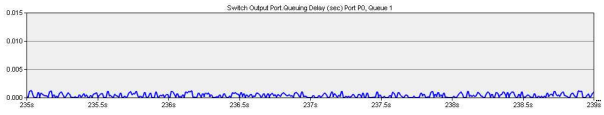
(a) Number of waiting bytes in output buffer in CoS2



(b) Number of waiting bytes in output buffer in CoS2



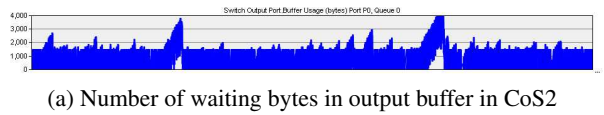
(c) Delay of Background traffic for crossing the switch 2 (in s)



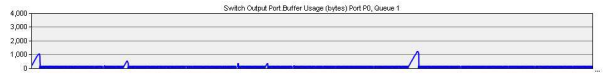
(d) Delay of Real-time traffic for crossing the switch 2 (in s)

Fig. 13. Network behavior with $w_1 = 141$ and $w_2 = 30$.

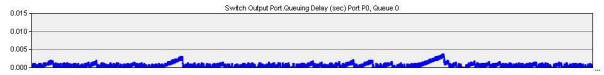
Without the fuzzy controller, Fig. 13 focuses the global simulation on only four seconds and collects the ensuing results. Fig. 13a and 13b give the number of bytes waiting in the output buffers of class 1 and class 2, respectively. Fig. 13c and 13d show the delays (in seconds) required to cross switch 2 for class 1 and class 2 traffic, respectively. The weights are fixed at $w_1 = 141$ and $w_2 = 30$. This configuration enables the network to maintain the frame delay of class 1 under the target delay value. The number of bytes waiting in the buffer of class 2 evolves based on the inter-arrival of background traffic (Fig. 13a). It is interesting to observe that the background traffic variation has no impact on the buffer of class 1 (Fig. 13b) and that the jitter (variation of delay) for the frames of class 1 is low. Fig. 13d shows that the frame delay of class 1 is under the threshold of 0.458 *ms*. Finally, Fig. 13c indicates that the frame delays of class 2 are impacted by their waiting bytes in the buffer. This result corresponds to an important drawback of this method (in addition to the network expertise level required). Because weights are fixed without considering the evolution of the background traffic, other traffics suffer from the large and potentially excessive reservation guaranteed to the real-time traffic.



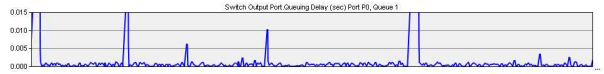
(a) Number of waiting bytes in output buffer in CoS2



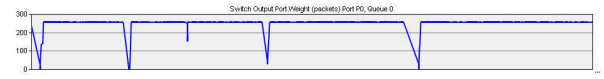
(b) Number of waiting bytes in output buffer in CoS2



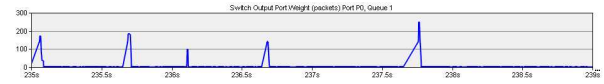
(c) Delay of Background traffic for crossing the switch 2 (in s)



(d) Delay of Real-time traffic for crossing the switch 2 (in s)



(e) Evolution of w_2



(f) Evolution of w_1

Fig. 14. Network behavior with fuzzy controllers.

Fig. 14 collects all of the simulation results when the fuzzy controller is activated. Fig. 14e and Fig. 14f show the evolution of the weights for w_2 and w_1, respectively. Fig. 14e highlights most of the time during which the fuzzy controller tries to offer the maximum bandwidth to the background traffic. Indeed, the w_2 value is near 255; although this corresponds to the maximum possible reservation, the tuning tends to cause accumulation in the buffer of class 1 (Fig. 14b) and generates peaks in the frame delay for class 1 (Fig. 14d). When this delay reaches intermediate or unacceptable delay areas, the controllers placed in the switches reconfigure the weights to offer more bandwidth to the frames of class 1. The consequence is that the frame delay for class 1 immediately returns to the target delay value. Therefore, this method successfully retrieves the QoS level delivered when no controllers are used and when a network expert only statically configures the weights.

The objective of the dynamic network controller

is to offer more bandwidth to low class traffic. When the background traffic sends 100 Mb/s, the dynamic configuration decreases to 6.67 s and the transfer time is compared to the static approach. This difference could be more important if the load of the high class traffic were greater.

These simulation results show the interest in dynamically tuning the network bandwidth relative to both the load variation and the application requirements. The results also indicate the fuzzy controller does not guarantee the high class traffic delay stays in the acceptable areas. The peaks in the delay in the unacceptable area are due to the nature of the simulated background traffic, which can arrive in a burst and is not regulated like the TCP flow on the Internet. Thus, the results observed are obtained from critical situations in which the network reactivity is tested under the worst cases. However, to avoid reaching the unacceptable area, the delay fuzzification model can be redesigned using lower delay thresholds between the target and intermediate areas and running the risk of generating starvation for the low class traffic. The tuning of these delay thresholds should be carefully studied according to the user specifications (i.e., hard and soft real-time systems). Another remark is that the scheduling policy is not preemptive, so the server continues the process making frames wait in the buffer before reconsidering a new weight tuning. Another solution for limiting the unacceptable area is applying the same approach described in this paper to another type of scheduler. One method that is generally implemented in the switch is mixing WRR with strict priority mechanisms like in IEEE 802.1 AVB.

6. Conclusion

The objective of this paper is to show the use of fuzzy logic in identifying and controlling a communication network. The paper addresses the hot topic of QoS management, in which tuning network parameters face non-linearities, as in the case of WRR scheduling in switched Ethernet architectures. The proposed approach consists of analyzing the network behavior from a numerical simulation benchmark. It uses the fuzzy classifier method to identify the system without using an analytic model and limiting the expertise steps. The model is obtained from the rules that are automatically learned by the classifier and is then utilized to design the controller. The recognition rates are relatively good considering the limited input data sets. Moreover, the Riverbed Modeler simulations show the fuzzy controllers distributed in the Ethernet switches are pertinent to control the end-to-end delays. Furthermore, it enables a framework of differentiated services to adapt the scheduling reservation to the real-time application needs while maintaining an acceptable network availability level for other applications. Hence, the same method could be applied both for switched Ethernet architectures and for networks relying on a differentiated services principle.

Finally, the approach developed in this paper is fully distributed. It could be interesting to define a network controller strategy by considering more global information such as the end-to-end delays or application performance.

References

1. R. Alcalá, J. Alcalá-Fdez, F. Herrera and J. Otero, Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation, *International Journal of Approximate Reasoning*, **44**(1) (2007) 45–64.
2. B. Bensaou, D.H.K. Tsang and King Tung Chan, Credit-based fair queueing (CBFQ): a simple service-scheduling algorithm for packet-switched networks, *IEEE/ACM Transactions on Networking*, **9**(5) (2001) 591–604.
3. M.R. Berthold, Mixed fuzzy rule formation, *International Journal of Approximate Reasoning*, **32**(2) (2003) 67–84.
4. V. Bombardier and E. Schmitt, Fuzzy rule classifier: Capability for generalization in wood color recognition, *Engineering Applications of Artificial Intelligence*, **23**(6) (2010) 978-988.
5. A. Chandrasekar and S. Arumugam, A Fuzzy Approach for Representative Node Selection in Cross Layer TCP, *Journal of Theoretical and Applied Information Technology*, **64**(1) (2014) 1–15.
6. O. Cordón, M.J. del Jesus and F. Herrera, A proposal on reasoning methods in fuzzy rule-based classification systems, *International Journal of Approximate Reasoning*, **20**(1) (1999) 21–45.
7. O. Cordón, F. Gomide, F. Herrera, F. Hoffmann and L.

Magdalena, Ten years of genetic fuzzy systems: current framework and new trends, *Fuzzy Sets and Systems*, **141**(1) (2004) 5–31.
8. F. de A.T. de Carvalho, Fuzzy -means clustering methods for symbolic interval data, *Pattern Recognition Letters*, **28**(4) (2007) 423–437.
9. A. Demers, S. Keshav and S. Shenker, Analysis and Simulation of a Fair Queueing Algorithm, *ACM SIGCOMM Comput. Commun. Rev.* **19**(4) (1989) 1–12.
10. M. Di Penta and L. Troiano, Using Fuzzy Logic to Relax Constraints in GA-Based Service Composition, in *Proc. of the conference on Genetic and Evolutionary Computation*, (2005).
11. D. Dubois and H. Prade, Fuzzy rules in knowledge-based systems – Modelling gradedness, uncertainty and preference, in *An Introduction to Fuzzy Logic Applications in Intelligent Systems*, Springer US, (1992) 45–68.
12. D. Dubois and H. Prade, What are fuzzy rules and how to use them, *Fuzzy Sets and Systems*, **84**(2) (1996) 169–185.
13. M.P. Fernandez, A. de Castro, P. Pedroza and J.F. de Rezende, Optimizing Fuzzy Controllers with Genetic Algorithms for QoS Improvement, in *International Telecommunications Symposium*, (2002).
14. M.P. Fernandez, A. de Castro, P. Pedroza and J.F. de Rezende, Converting QoS policy specification into fuzzy logic parameters, *Teletraffic Science and Engineering*, **5** (2003) 339–348.
15. A. Fithritama, E. Rondeau, V. Bombardier and J.-P. Georges, Modeling fuzzy rules for managing power consumption of Ethernet switch, in *23rd International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, (2015) 42–47.
16. J. Forest, M. Rifqi and B. Bouchon-Meunier, Class Segmentation to Improve Fuzzy Prototype Construction: Visualization and Characterization of Non Homogeneous Classes, in *IEEE International Conference on Fuzzy Systems*, (2006) 555–559.
17. S. Gajjar, M. Sarkar and K. Dasgupta, Fuzzy-Cross: A fuzzy based architecture for wireless sensor network, *Journal of Computational Methods in Sciences and Engineering*, **15**(4) (2015) 801–823.
18. G. Feng, A Survey on Analysis and Design of Model-Based Fuzzy Control Systems, *IEEE Transactions on Fuzzy Systems*, **14**(5) (2006) 676–697.
19. J.-P. Georges, T. Divoux and E. Rondeau, Confronting the performances of a switched Ethernet network with industrial constraints by using the network calculus, *International Journal of Communication Systems* **18**(9) (2005) 877–903.
20. A. Hamam, M. Eid, A. El Saddik and N.D. Georganas, A Fuzzy Logic System for Evaluating Quality of Experience of Haptic-Based Applications, in *Haptics: Perception, Devices and Scenarios: 6th International Conference, EuroHaptics*, (2008) 129–138.
21. P.-Y. Hao, J.-H. Chiang and Y.-K. Tu, Hierarchically SVM classification based on support vector clustering method and its application to document categorization, *Expert Systems with Applications*, **33**(3) (2007) 627–635.
22. IEEE Standard for Local and Metropolitan Area Networks–Audio Video Bridging (AVB) Systems, In *IEEE Std 802.1BA-2011*, (2011) 1–45.
23. IEEE Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP), In *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)*, (2010) 1–119.
24. IEEE Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams, In *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, (2010) C1–72.
25. H. Ishibuchi, K. Nozaki and H. Tanaka, Distributed representation of fuzzy rules and its application to pattern classification, *Fuzzy Sets and Systems*, **52**(1) (1992) 21–32.
26. H. Jones, Raisonnement à base de règles implicatives floues – Inférence et Sémantique, PhD Thesis, University of Toulouse (France), 2007 (in french).
27. T. Kempowsky, A. Subias and J. Aguilar-Martin, Process situation assessment: From a fuzzy partition to a finite state machine, *Engineering Applications of Artificial Intelligence*, **19**(5) (2006) 461–477.
28. L. Khoukhi and S. Cherkaoui, Experimenting with fuzzy logic for QoS management in mobile ad hoc networks, *International Journal of Computer Science and Network Security*, **8**(8) (2008) 372–386.
29. S. Kubler, W. Derigent, E. Rondeau and A. Thomas, Embedding Information on Communicating Materials from Context-Sensitive Information Analysis Based on Fuzzy AHP Theory, in *IEEE International Conference on Green Computing and Communications*, (2012) 240–243.
30. H.-T. Lim, D. Herrscher, M.J. Waltl and F. Chaari, Performance Analysis of the IEEE 802.1 Ethernet Audio/Video Bridging Standard, in *Proc. of the 5th International ICST Conference on Simulation Tools and Techniques*, (2012) 27–36.
31. L. Lo Bello, G. A. Kaczynski and O. Mirabella, Improving the real-time behavior of Ethernet networks using traffic smoothing, *IEEE Transactions on Industrial Informatics*, **1**(3) (2005) 151–161.
32. J. M. Mendel, Fuzzy logic systems for engineering: a tutorial, *Proceedings of the IEEE*, **83**(3) (1995) 345–377.
33. D. Michie, D. Spiegelhalter and C. Taylor, *Machine Learning: Neural and Statistical Classification*, Ellis

Horwood, (2009).
34. M. Mohan Rao and V. Krishna, Fuzzy based intelligent routing in high speed networks, *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, **3**(10) (2014) 3372–3379.
35. D. Nauck and R. Kruse, NEFCLASS-X — a Soft Computing Tool to Build Readable Fuzzy Classifiers, *BT Technology Journal*, **16**(3) (1998) 180–190.
36. D. Nauck and R. Kruse, Obtaining interpretable fuzzy classification rules from medical data, *Artificial Intelligence in Medicine*, **16**(2) (1999) 149–169.
37. K. Nozaki, H. Ishibuchi and H. Tanaka, A Simple but Powerful Heuristic Method for Generating Fuzzy Rules from Numerical Data, *Fuzzy Sets Syst.*, **86**(3) (1997) 251–270.
38. R.-E. Precup and H. Hellendoorn, A survey on industrial applications of fuzzy control, *Computers in Industry*, **62**(3) (2011) 213–226.
39. N. A. M. Radzi, N. M. Din, M. S. A. Majid and M. H. Al-Mansoori, Global priority DBA using fuzzy logic in Ethernet PON, in *The 17th Asia Pacific Conference on Communications*, (2011) 113–116.
40. J.A. Roubos, M. Setnes and J. Abonyi, Learning fuzzy classification rules from labeled data, *Information Sciences*, **150**(1) (2003) 77–93.
41. P.H. Rusmin, Internet Congestion Control Using Fuzzy Integral Controller, *International Journal on Electrical Engineering and Informatics*, **6**(3) (2014) 497–510.
42. E. Schmitt, V. Bombardier and P. Charpentier, Self-Fuzzification Method according to Typicality Correlation for Classification on tiny Data Sets, in *IEEE International Fuzzy Systems Conference*, (2007) 1-6.
43. S.P. Setti, D.V. Vijay Kumar, G.S.M, Nagendra Prasad and K. Narasimha Raju, Implementation of Fuzzy Priority Scheduler for MANET and Performance Analysis with Reactive Protocols, *International Journal of Engineering Science and Technology*, **2**(8) (2010) 3635–3640.
44. K. Steinhammer, P. Grillinger, A. Ademaj and H. Kopetz, A Time-Triggered Ethernet (TTE) Switch, in *Proc. of the Design Automation & Test in Europe Conference*, (2006) 1–6.
45. S. Suardinata and K. Abu Bakar, A fuzzy logic classification of incoming packet for VOIP, *TELKOMNIKA Telecommunication, Computing, Electronics and Control*, **8**(2) (2010) 165–174.
46. M. Sugeno, An introductory survey of fuzzy control, *Information Sciences*, **36**(1) (1985) 59–83.
47. D. Todinca, H. Graja, P. Perry and J. Murphy, Novel admission control algorithm for GPRS/EGPRS based on fuzzy logic, in *Fifth IEE International Conference on 3G Mobile Communication Technologies*, (2004) 342-346.
48. J.Y. Tou, Y.H. Tay and P.Y. Lau, Recent trends in texture classification: a review, in *Symposium on Progress in Information & Communication Technology*, (2009) 63–68.
49. U. Urathal alias Swathiga and C. Chandrasekar, An Efficient Fuzzy based Congestion Control Technique for Wireless Sensor Networks, *International Journal of Computer Applications*, **40**(14) (2012) 47-55.
50. C. Venkatesh, N. Yadaiah and A.M. Natarajan, Dynamic source routing protocol using fuzzy logic concepts for ad hoc networks, *Academic Open Internet Journal*, **15** (2010).
51. L. X. Wang and J. M. Mendel, Generating fuzzy rules by learning from examples, *IEEE Transactions on Systems, Man, and Cybernetics*, **22**(6) (1992) 1414–1427.
52. F. Xia, W. Zhao, Y. Sun and Y.-C. Tian, Fuzzy Logic Control Based QoS Management in Wireless Sensor/Actuator Networks, *Sensors*, **7**(12) (2007) 3179–3191.
53. L.A. Zadeh, Fuzzy sets, *Information and Control*, **8**(3) (1965) 338–353.
54. L. A. Zadeh, Outline of a New Approach to the Analysis of Complex Systems and Decision Processes, *Transactions on Systems, Man, and Cybernetics*, **SMC-3**(1) (1973) 28–44.
55. L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning–I&II, *Information Sciences*, **8**(3) (1975) 199–249, 301–357.