Adaptation of MAPE-K and Fuzzy Q-Learning in SLA management

Ahmad Kamal Ramli, Mohd Helmy Abd Wahab, Syed Zulkarnain Syed Idrus
Faculty of Computer Science and Information Technology
Selangor International Islamic University College (KUIS)
Faculty of Electrical and Electronic Engineering
Universiti Tun Hussein Onn Malaysia, Johor, Malaysia
School of Communication and Human Development
Universiti Malaysia Perlis, Perlis, Malaysia
ahmadkamal@kuis.edu.my

Abstract. A Service Level Agreement (SLA) is the legal catalyst to monitor any contract violation between end users and ISPs and is embedded within a Quality of Service (QoS) framework. The key to the proposed architecture is the utilization of self-capabilities designed to have self-management over uncertainties and the provision of self-adaptive interactions. Thus, the Monitor, Analyse, Plan, Execute and Knowledge Base (MAPE-K) approach can deal with this problem together with the integration of Fuzzy and Q-Learning algorithms. The proposed experiment is in the context of autonomic computing. An adaptation manager is the main proposed component to update admission control on the ISP current resources and the ability to manage SLAs. The proposed solution, demonstrating Q-Learning works adaptive with QoS parameters, e.g. Latency, Availability and Packet Loss. With the combination of fuzzy and Q-Learning, we demonstrate that the proposed adaptation manager is able to handle the uncertainties and learning abilities. Q-Learning is able to identify the initial state from various ISPs iterations and update them with appropriate actions, reflecting the reward configurations. The higher the iterations process the higher is the increase the learning ability, rewards and exploration probability.

Keywords. Service Level Agreements, Quality of Service, Internet Service Provider, MAPE-K, Autonomic Computing , Reinforcement Learning and Fuzzy

1. Introduction
QoS is the set of service requirements to be fulfilled by the network providers in relation to delivering guaranteed services during network activities. In the recent world, half of the global population [1] are actively connected to the Internet and further initiatives have been made by Facebook, Motorola, Nokia, etc. [2] to capitalise the offline users with equal Internet connectivity through community service responsibility. Social engineering software, Voice Over Internet Protocol (VOIP), Instant Messaging, online shopping, video streaming and robust innovations in communication devices are the major causes of why people heavily connect to the Internet and experience bandwidth issues.
The customer charges and service performance vary from one ISP to another because of the different rates, robust business model and inter-ISPs establishment fees. Yet, ISPs have the freedom to pick out and switch any of the associates who have provided them with good Internet quality or at least, best-effort services. An SLA Manager is the element that is available in the ISP architecture to accept or reject offers based on the correct information supplied by admission control. An SLA framework alongside monitoring QoS provision will be a major contribution towards the solution.

2. Background

The Internet has been fluctuating robustly within the last ten years. Especially with the introduction of appliances and gadgets. Creative people planned for the modern world. Small devices dynamically change the nature of people using communication via the Internet rather than conventional circuit connectivity. Skype’s current feature allows people to receive a real time translation when communication is engaged with two people with different and distinct native languages. Furthermore, the utmost competition between smart phone makers will rapidly transform peoples’ behaviour in adjusting to the technological revolution.

2.1 Quality of Service (QoS)

QoS specification is concerned with capturing the application’s level quality of the service requirements and management policies. The QoS specification is generally different in each system layer and it is used to configure and maintain the QoS mechanism residents in each layer. For example, at the distributed system platform level, the QoS specification is primarily user-oriented rather than system-oriented. Lower level considerations such as tightness, the synchronisation of multiple related flows, the rate and burst size of flows, or the detail of thread scheduling should all be hidden at this level.

2.2 SLA

SLA was introduced back in the 1980s by telecommunication companies. By giving an SLA, the telecommunication providers can spell out their commitment to the customer either it as a best-effort connection or guaranteed connection. Normally, the customer will trust the ISP commitment by saying that their services can do almost everything before the establishment of any agreement. The subscription can either be broadband or a leased line depending upon the needs of the user. Other issues are to do with the IPv4 running out, and some countries have established IPv6 as their pillar. Lots of Internet activities such as Cloud computing and other services have a SLA as a must between subscribers and the providers to ensure that the services are according to their promises. Starting in 2011, we can see the parallel contributions made by researchers and industrial players to identify possibilities to address this situation.

2.3 Adaptive Architecture

The concept consists of the iterative process to ensure the sequential flow from one to another when improvising. Basically, in learning, there are two routes; cognitive and adaptive learning. Cognitive is known to be a static method and the user or the learner should enhance their own capabilities without any intelligent interference from the application or system that they are using at the moment.

Adaptive will understand the user's capabilities and will afterwards assist him/her in areas that he should improve. The system itself is capable of recording and tracking his progress individually and next it can identify the most relevant way to improve his or her counting skills. In the computer network, this method is used to understand the current workloads that are coming from various sources and it will mitigate and normalise this into the available resources.
2.3.1 Learning Technique

Artificial intelligence is the discipline that used an appropriate machine learning method to capture human intelligence and later translate to machine codes. To some extent, into the same ability within a computer. It can be improved, in the logical sense, by creating an application with artificial intelligence elements, such as neural networks, fuzzy logic, a decision support system, knowledge base, etc [6]. However, this method still requires human intervention in understanding their right and false alarms before it can really confirm the attended situation. Supervised, semi-supervised, non-supervised and reinforcement learning are the methods available in Machine Learning to teach agent with human behaviour and gradually increases their learning ability.

2.3.2 Dynamic Network Reconfiguration

The proposed autonomic computing can manage the framework and free the system administrator from routine tasks in the networking environment. Moving towards IBM in self-management embraces four elements.

A. Self-Configuration
In conventional computing, this is done by corporate data centres that have multiple vendors and platforms. Installing, configuring, and integrating systems is time consuming and error prone. However, in autonomic computing, the automated configuration of components and systems follows high level policies. The rest of the system components adjust automatically and seamlessly.

B. Self-Optimisation
This feature ensures that the components and systems continually seek opportunities to improve their own performance and efficiency. This is difficult in current computing where systems have hundreds of manually set, non-linear tuning parameters and their number increases with each release.

C. Self-Healing
Having this opportunity, autonomic computing can automatically detect, diagnose and repair localised software and hardware problems. This is problematic in conventional approaches due to problem determination. In large, complex systems that can take a team of programmer’s weeks.

D. Self-Protection
In current practise, the detection of and recovery from attacks and cascading failures is manual. Placing this feature will ensure that the system automatically defends itself against malicious attacks or cascading failures. It uses early warnings to anticipate and prevent system-wide failures.

Schroeder [7] introduced Autonet, to forecast the possibility of handling huge high-speed data transactions in a next-generation approach. However, in the same year, J.M. Garcia and J. Duato investigated [8] dynamic reconfiguration and focused on transparent processing between one network node to another. The novelty of that research was to ensure that the flow of data runs smoothly and is less affected by parallel applications within the network. In contrast, the approach was limited to one aspect of self-configuring, and it is not an inclusive model.

Several researchers have presented significant work on autonomic management from various aspects. The vast knowledge base has been injected into research to justify the flexibility of taxonomy and semantic searching capabilities that can drive the autonomic management forward into better decision-making [9]. The outcome was very positive, whereby self-management was capable to understand the content of a predefined knowledge base. To extend the capabilities of autonomic management to intra-domain capabilities, research in [4] was successful in implementing an algorithm in a test bed environment.
2.3.3 Adaptive Management

Two main contributions that are significant in the scope of this paper are self-management with the ability to support SLAs and managing resources using autonomic management.

In the Self-Adaptive literature, research conducted by [10] focused on adaptive pricing strategies with the assumption of healthy competition between domain operators to get as much inter-domain traffic profit as possible. The competition was based on the available route connections, prices, agents and the network load. The later research by [15] expressed the adaptation framework with the integration of Fuzzy Q-Learning to handle the varied workload performance from cloud providers.

The learning methods were able to produce the learning rate of each performance. However, the number of inputs was limited to two inputs and the iterations numbers were on a small scale. The same research has been enhanced with back to back comparison with Q-Learning and Sarsa to understand each capacity in relation to handling the workload.

Besides Fuzzy Q-learning, Finite Action Learning Automata (FALA) and Continues Reinforcement Learning Automata (CARLA) have been applied by [96] in his research. The objectives were to demonstrate that Interdomain model routing, as an eternal flow on a certain link, has a cost for the ISP owner. Interdomain links are shared among domains and their prices are equal to the routing costs. They provide virtual link prices to generate income. The pricing model with the appropriate costs and utilities has been introduced as the outcome of his work.

In the fuzzy system, the research made by [11-12] demonstrates the benefit of the adaptation of the available services together with SLA. The adaptation also helps to maximise profits in service level management [13], deals with the negotiation [11] and lastly, the adaptation is based on QoS requirements [14].

3. Proposed Research

To meet this target, there are FIVE (5) objectives that have been established and identified, which include:

I. Exploring the current research and issues related to adaptive framework, autonomous computing, QoS, SLA, ISP and Machine Learning. This is here to help understand the current progress and how the remaining progress work can be established in this research. It provides information on the engagement of the connected domains for them to be unified, so they are able to be a solid tool for the execution of this study.

II. Investigating the adaptive framework, SLA, and available case studies. This is to understand the ability of the adaptive framework and match that is within the available information which is publicly available for ISP case studies on their performance and connectivity.

III. Exploring the Telco, ISP, and admission control architecture together with the SLA Manager process. This work contributes to the dynamic understanding of each component and how the SLA manager manages the SLA process.

IV. Investigating the fuzzy systems and Machine Learning algorithms to provide an adaptive mechanism. This activity is used to understand the exact combination between the fuzzy system and machine learning. The ideal combination can handle the uncertainties and learning abilities during the iteration and provides dynamic adaptations.

V. Exploring simulation software to predict on the implementation of this research. Simulation software is vital to this research because it provides the evidence for the mathematical algorithm execution, and for the discrete and network simulation software.

4. Experiment

In Q-Learning, it is derived from Markov Decision Process (MDP) and consists of three inputs. State, action and reward. Every state will have a different score depending on the steps taken toward the final policy and continuing with the learning update.
The algorithm will have an iteration process. This is known as an episode to ensure that it will optimise with the alpha and gamma values:

a) \( \gamma = \text{Gamma} \) value in the range of 0 and 1. The lowest value will instruct Q-Learning to find the instance reward and ignore the total score of the accumulated reward.

b) \( \alpha = \text{Alpha} \) value identical to gamma, which is a range between 0 and 1. In normal circumstances, the value is set low to optimise the algorithm, such as 0.2.

**Definition 1. Q-Learning Update**

\[
Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_{t+1} + \gamma^\max Q(s_{t+1}, a) - Q(s_t, a_t)]
\]

**Q-Learning Algorithm [131]**

| Initialize \( Q_0 (s, a) \) to random values |
| Choose a starting point \( s_0 \) |
| While the policy is not good enough |
| Choose at according to values \( Q_t (s_t, .) \) |
| \( at = f(Q_t(s_t, .)) \) |
| Obtain in return: \( s_t + 1 \ (s') \) and \( r_t \) |
| Update using Definition 13 |
| End While |

4.1 Fuzzy Q-Learning

This is the algorithm based on Q-Learning and the limitations of fuzzy, introduced by [17] and iteratively enhanced by [18], [16] and [19]. The constraint of fuzzy [20], is that it must be heavier that ‘W’, where in his case the ‘W’ refers to Weight. On the other hand, Fuzzy Q-Learning actions can process fuzzy constraints and proceed with the optimal policy. This feature is not available for the actions in Q-learning introduced by [17].

4.2 Algorithm

This section presents an approach which provides the overall algorithm of this research. The algorithm itself is segmented into phases and produces the results for further analysis.

**Implementation of the Complete Algorithm (Excerpt)**

**Phase One.**

1. Acceptance of the three QoS parameters (Latency, Availability and Packet loss).

**Phase Two**

1. Execution of QoS parameters from case study [21] (Pass, Low, Normal, High, Peak) to centre of weighted approach.
   - Use scores for each performance
   - Use reward for Q-Learning algorithm
   - Populate the values for next phase

**Phase Three**
1. Application of fuzzy Sugeno for membership functions and rule base

4.3 Experimental Configuration
In this approach, there are three scenarios of QoS comparison which are:
I. Latency versus Availability
II. Packet Loss versus Availability
III. Packet Loss vs Latency.

In the each of the comparisons, the number of iterations was fixed to 500 and a total of 125 rule base combinations were identified to cater for all possible ranges of the selected QoS parameters.

Table 4.1. Rule base declaration in Fuzzy Toolbox (Excerpt)

| Rule | QoS Parameters | Scores group on the service Performance | Weighted Total using centre of weighted average method |
|------|----------------|----------------------------------------|-----------------------------------------------------|
|      | Latency        | Availability                           | Packet Loss | -2 | -1 | 0 | 1 | 2 |                               |
| 1    | Low            | Pass                                   | Low         | 0  | 0  | 0 | 2 | 1 | 2.00                          |
| 2    | Low            | Pass                                   | Normal      | 0  | 0  | 1 | 1 | 1 | 1.50                          |
| 3    | Low            | Pass                                   | High        | 0  | 1  | 0 | 1 | 1 | 1.00                          |
| 4    | Low            | Pass                                   | Peak        | 1  | 0  | 0 | 1 | 1 | 0.50                          |
| 5    | Low            | Pass                                   | Pass        | 0  | 0  | 0 | 1 | 2 | 2.50                          |
| 6    | Low            | Peak                                   | Low         | 1  | 0  | 0 | 2 | 0 | 0.00                          |
| 7    | Low            | Peak                                   | Normal      | 1  | 0  | 1 | 1 | 0 | -0.50                         |
| 8    | Low            | Peak                                   | High        | 1  | 1  | 0 | 1 | 0 | -1.00                         |

The next phase is the purification of 125 rule base into five groups which are Low, Normal, High, Peak and Pass. The total of the 25 rules base represents the composition of the overall rules that have been identified and match with the values calculated with the centre of the weighted average (Equation).

Table 4.2. Rule base in the State values (Latency versus Availability) (Excerpt)

| Latency | Availability | Rules sequence in Matlab | Values according to Rules |
|---------|--------------|--------------------------|---------------------------|
| Low     | Pass         | 1                        | Mf8                       |
| Low     | Peak         | 2                        | Mf4                       |
| Low     | High         | 3                        | Mf5                       |
| Low     | Normal       | 4                        | Mf6                       |
| Low     | Low          | 5                        | Mf7                       |

Table 4.3. Membership function values – Centre of weighted average (Latency versus Availability)

| Membership Function | Values |
|---------------------|--------|
| Mf1                 | -2     |
| Mf2                 | -1.5   |
| Mf3                 | -1     |
| Mf4                 | -0.5   |
We applied the Sugeno approach using fuzzy toolbox to handle the type-2 fuzzy values which are available in MATLAB. This is the first part of the fuzzy approach up to the inclusion of defuzzification values, which were later combined in Q-Learning.

The reward approach was applied with positive and negative values. The justification of each value was identified accordingly to the grouping of the QoS parameters. Tables 4.4-4.7 tabulates the information accordingly.

**Table 4.4:** Linguistic variables for the negative reward

| Description | Category | Values          |
|-------------|----------|-----------------|
| Latency     | Low      | 500 ms ≤ La < 750 ms |
|             | Normal   | 750 ms ≤ La < 1 s  |
|             | High     | 1 s ≤ La < 5 s    |
|             | Peak     | 5 s ≤ La          |
| Availability| Low      | 95% < Av ≤ 98%    |
|             | Normal   | 90% < Av ≤ 95%    |
|             | High     | 80% < Av ≤ 90%    |
|             | Peak     | Av ≤ 80%          |
| Packet Loss | Low      | 2% ≤ PL < 4%      |
|             | Normal   | 4% ≤ PL < 8%      |
|             | High     | 8% ≤ PL < 20%     |
|             | Peak     | 20% ≤ PL          |

**Table 4.5.** Positive Reward

| Inputs       | Reward | Minimum | Mean | Maximum |
|--------------|--------|---------|------|---------|
| Latency      | Positive | 499 ms  | 249  | 0 s     |
| Availability | Positive | 99 %    | 99.5 %| 100%    |
| Packet Loss  | Positive | 1 %     | 0.5 %| %       |

**Table 4.6.** Scores for the service of the performance.

| Description | Score |
|-------------|-------|
| Peak        | -2    |
| High        | -1    |
| Normal      | 0     |
| Low         | 1     |
| Pass        | 2     |

**Table 4.7:** Group Rewards

| Group | Score |
|-------|-------|
| Low   | -1    |
Table summarises the executions of experiments. As for the Q-Learning part, it is the extension of fuzzy values and has three Q-learning factors. The runs evaluated the groups of Small, Medium and Large of the Learning Rate and Small and Medium of Epsilon and Lambda.

### Table 4.8. Categories of the experiments (Excerpt)

| Categories                  | No of iterations | Type of Comparison |
|-----------------------------|------------------|--------------------|
| Latency vs Ability          | 500              | Small Epsilon and Small Lambda |
|                             |                  | Small Learn Rate   |
|                             |                  | Medium Learn Rate  |
|                             |                  | Large Learn Rate   |
| Large Epsilon and Large Lambda |                | Small Learn Rate   |
|                             |                  | Medium Learn Rate  |
|                             |                  | Large Learn Rate   |

Below are the values for the categories of epsilon, lambda and alpha.

### Table 4.9. Categorisation of the Q-Learning Factors

| Q-Learning Factor | Categories | 0.1 |
|-------------------|------------|-----|
| Epsilon           | Small      | 0.1 |
|                   | Large      | 1.0 |
| Lambda            | Small      | 0.1 |
|                   | Large      | 1.0 |
| Learn Rate ( Alpha) | Small    | 0.1 |
|                   | Medium     | 0.5 |
|                   | Large      | 1.0 |

The final step was to update the correct reward values and to ensure that the conditions were according to the rule base values.

#### Reward Calculator Embedded within MATLAB Code (Excerpt)

```matlab
if (current_state(1)>= 0.5 && current_state(1) <0.75 )|| (current_state(2) > 95 && current_state(2) <=98)
    reward =-1;
    % Latency Normal
elseif (current_state(1)>= 0.75 && current_state(1) <1 )|| (current_state(2) > 90 &&
    current_state(2) <=95)
    reward =-2;
    %Latency High
elseif (current_state(1)>= 1 && current_state(1) <5 )|| (current_state(2) > 80 &&
    current_state(2) <=90)
end
```
reward =-3;
% Latency Peak
elseif (current_state(1)>= 5 )|| (current_state(2) < 80)
reward =-4;
% Latency Pass
elseif (current_state(1)<=0.5 )|| (current_state(2) > 98)
reward =1;
% if (current_state(2)<=SLA(1)) % response time SLO has not been violated
% reward=1;
elseif (current_state(2)<=old_state(2))&&
(old_action>0) ||
(current_state(1)<=old_state(1))&&(old_action>0) % violated but has been improved due to the
action
reward=0; % not either penalize nor give positive reward
% else %violated and has been dropped from last time
%reward=exp((98-curren_t_state(2))/98-1; % a negative reward between [-1 0]
%reward = -1;
%end
end
end

5. Result

5.1 Uncertainties
The ability of fuzzy logic to react to the robust values of QoS parameters, which are associated with
the defined rule base. The learning rate was then calculated from the end process of defuzzification
using the fuzzy toolbox. The change in the learning rate occurred when the input of the QoS
parameters reflected the linking rule.

i. Learning Rate with the Latency = 5 and Availability = 85
ii. Learning Rate with the Latency =2.25 and Packet Loss = 59.1
iii. Learning Rate with the Packet Loss = 80.5 and Availability = 76.4
The values of the learning rate changed to negative due to the poor performance of the QoS
parameters in the static inputs. Showed that the Latency was 2.25 and Packet Loss was 59.1, which
meant that the learning rate was -1.5. This is due to the Latency itself, which was high and the Packet
Loss was at its Peak. In this run, it activated rule 14 in the fuzzy inference system.

Another combination of Packet Loss, which is 80.5, and Availability is 76.4. This totalled to -2. The
learning rate is different in every combination and produced both negative and positive learning
abilities.
Throughout the experiment, there were no available rules fired for Packet Loss and Latency using the
data file. Therefore, the results are limited to two combinations, which are Latency versus Availability
and Availability versus Packet Loss. The outcome for the Small learning rate shows that most of the
values are still negative and that the learning rate performance increasing to positive with a higher
learning rate and ability to handle uncertainties.

5.2 Adaptation
This is the key output for the adaptation approach. It simply monitors, analyses, plans and executes
within the iteration. There are groups of iterations such as 100, 200, 300, 400 and 500. The reason for
doing this is to demonstrate the adaptation features from this experiment to deal with the input file.
SLA has been defined within the combination of QoS parameters in the rule base.
As for the overall adaptation in Packet Loss and Availability, the number of rules affected has been labelled with state, rule base as the action and lastly, q-values. There were 6 cases used to measure the overall adaptation and the evolution of the q-values associated to each action. The values of Small, Medium and Large have been defined.

Evaluation of the QoS Parameters:

i. Packet Loss versus Availability
   The evolution of the q-values increased from case 1 to case 3 and then dropped at case 4. The value continued growing from case 5 to case 6.
   - Case 1 – Small [Epsilon and Small Lambda] versus Small Learn Rate

ii. Latency versus Availability
   The overall adaptation for this combination shows that from case 2 up to the end of case 3, the q-values increase with negative rewards and the output started declining from case 4. However, the q-values continue rising from case 5 until case 6.

Conclusion

In the last experiment, it was an extension of second experiment which highlighted uncertainties and the ability of MAPE-K to handle the adaptation together with the self-learning showcase. Fuzzy Q-Learning was the adopted algorithm for this experiment. According to the generated results from the public data files and SLA case studies, the results have been segmented into three QoS combinations. The THREE (3) QoS parameters are Packet Loss, Availability and Latency, and these have been further categorised into different sets of Q-Learning factors. The analysis of the results are:

I. Packet Loss versus Availability

II. Latency versus Availability
   The analysis was applied for both Packet Loss versus Availability and Latency versus Packet Loss. The experiments were conducted with SIX (6) cases and the overall adaptation result was that the lowest Learn Rate produced the lowest Q-values. Although the same high learn rate was applied in both case 3 and case 6, the q-values in case 6 grew dramatically due to the higher values of Lambda and Epsilon in case 6. To be exact, the values in case 3 for Epsilon and Lambda were both 0.1, whereas in case 6, both were 1.0. The higher the value from 0.1 towards 1.0 represents the exploration and better reward function in the q-learning algorithm.

III. Latency versus Packet Loss
   In this exercise, there were no match rules concerning using the public access input files.

The outcome of these three experiments was to prove that the adaptation of self-learning utilising a Fuzzy Q-Learning algorithm can address the third research question. It is self-explanation as to how the different q-learning factors react accordingly to the accumulated learning rate, rewards and exploration.

6. References

[1] US Census Bureau, InternetWorldStats. CNNIC, Tencent, Facebook, ITU, CIA . “ Global Digital Statistics 2014 [ Online ] . Available : http://wearesocial.sg/
[2] [Online]. Available : http://www.internet.org/
[3] Ezhilchelvan, P. D., & Shrivastava, S. K. (2004). A model and a design approach to building QoS adaptive systems. In Architecting Dependable Systems II (pp. 215-238). Springer, Berlin, Heidelberg.
[4] M. Femminella, R. Francescangeli, and G. Reali, “An Enabling Platform for Autonomic Management of the Future Internet An Enabling Platform for Autonomic Management of the.”
[5] Autonomic Computing, White Paper, “An Architectural blueprint for autonomic computing .”, IBM, Third Edition, June 2005.
[6] Xiao-Jing Li, Bo Zhou, Jin Xiang Dong, “Self Learning Histograms for changing workloads”, Database Engineering and Application Symposium, 2005. IDEAS 2005. 9th International, pp. 229-234, IEEE, July 2005.

[7] Michael D. Schroeder, Andrew D. Birrell, Michael Burrows, Hal Murray and Roger M. Needham, “Autonet: A High Speed, Self-Configuring Local Area Network Using Point to Point Links”, IEEE Journal On Selected Areas In Communication, vol 9, no. 8, pp. 1318-1335, October 1991.

[8] J.M. Garcia and J. Duato, “An Algorithm for Dynamic Reconfiguration of a Multicomputer Network”, In Proceeding of the Third IEEE Symposiums on Parallel and Distributed Processing 1991, December 1991, pp. 848 – 855.

[9] Brendan Jennings, Sven van der Meer, Sasitharan Balasubramaniam, Dmitri Botvich, Michael O Foghlu and William Donnelly, “Towards Autonomic Management of Communications Networks”, IEEE Communications Magazine, Vol. 45, No. 10, pp. 112-121, October 2007.

[10] Vranx, P., Gurzi, P., Rodriguez, A., Steenhaut, K., & Nowé, A. (2015). A reinforcement learning approach for interdomain routing with link prices. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 10(1), 5.

[11] Kohli, Manbeen. "An Enhanced Goal-Oriented Decision-Making Model for Self-Adaptive Systems." (2011).

[12] Defazio, Aaron, and Thore Graepel. "A comparison of learning algorithms on the arcade learning environment." arXiv preprint arXiv:1410.8620 (2014).

[13] Brys, Tim, et al. "Policy transfer using reward shaping." Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[14] Pulakka, Kimmo. "A dynamic control system for adjusting prices and quality of service in DS enabled networks." Network Control and Engineering for QoS, Security and Mobility. Springer US, 2003. 241-252.

[15] Jamshidi, Pooyan, et al. "Fuzzy Self-Learning Controllers for Elasticity Management in Dynamic Cloud Architectures." 2016.

[16] Glowaty, G. (2013). Enhancements of fuzzy Q-learning algorithm. Computer Science, 7(4), 77.

[17] Dayan, P., & Watkins, C. J. C. H. (1992). Q-learning. Machine learning, 8(3), 279-292.

[18] Pierre Yves Glorennec and Lione Jouffe: Fuzz Q-Learning. Proc. Sixth IEEE International Conference on Fuzzy Systems (Fuzz-IEEE’97), Barcelona 1997, pp. 659-662

[19] Caironi, P. V., & Dorigo, M. (1997). Training and delayed reinforcements in Q-learning agents. International journal of intelligent systems, 12(10), 695-724.

[20] Pedrycz, Witold, and Fernando Gomide. An introduction to fuzzy sets: analysis and design. Mit Press, 1998.

[21] SLA Case Study IRAN. Retrieved from https://www.apt.int/sites/default/files/Upload-files/SATRC/WGNET02/SATRC-WG-NET02-08_Service_Level_Agreements.pdf