A Reinforcement Learning QoS Negotiation Model for IoT Middleware

Itorobong S. Udoh and Gerald Kotonya
School of Computing and Communications, Lancaster University, Bailrigg, Lancaster, U.K.

Keywords: QoS, Internet of Things, Negotiation, IoT Services, Reinforcement Learning.

Abstract: A large number of heterogeneous and mobile devices interacting with each other, leading to the execution of tasks with little human interference, characterizes the Internet of Things (IoT) ecosystem. This interaction typically occurs in a service-oriented manner facilitated by an IoT middleware. The service provision paradigm in the IoT dynamic environment requires a negotiation process to resolve Quality of Service (QoS) contentions between heterogeneous devices with conflicting preferences. This paper proposes a negotiation model that allows negotiating agents to dynamically adapt their strategies using a model-based reinforcement learning as the QoS preferences evolve and the negotiation resources changes due to the changes in the physical world. We use a simulated environment to illustrate the improvements that our proposed negotiation model brings to the QoS negotiation process in a dynamic IoT environment.

1 INTRODUCTION

The IoT ecosystem emerges from a core of a heterogeneous mix of several technologies. The identification, sensing, communication and middleware technologies form the fundamental building blocks required to incorporate "intelligence" into "things" (Patel and Patel, 2016). Several research initiatives have structured these IoT enabling technologies into *n*-layered technology architectures. Most of the proposed technology architectural models add more abstractions to the primary 3-layer architecture, which consists of the *application/service*, *communication/network* and *perception/physical* layers as illustrated in Figure 1.

Alongside these layers, it is important to address QoS management concerns to avoid situations that may lead to serious problems especially in IoT systems that are characterised by stringent QoS needs such as embedded IoT medical devices and autonomous vehicle control systems. Beyond these are many IoT systems for which providing best-effort QoS may not be adequate for the successful operation of such systems.

Guaranteeing the QoS requirements demands that all the layers in the IoT architecture provide both effective and efficient QoS management strategies (Duan, Chen and Xing, 2011). A recent survey by

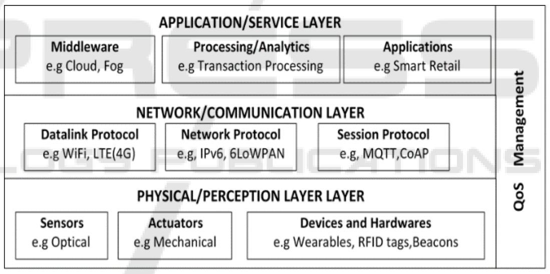
APPLICATION/SERVICE LAYER
Middleware e.g Cloud, Fog
Processing/Analytics e.g Transaction Processing
Applications e.g Smart Retail
NETWORK/COMMUNICATION LAYER
Datalink Protocol e.g WiFi, LTE(4G)
Network Protocol e.g, IPv6, 6LoWPAN
Session Protocol e.g, MQTT,CoAP
PHYSICAL/PERCEPTION LAYER LAYER
Sensors e.g Optical
Actuators e.g Mechanical
Devices and Hardwares e.g Wearables, RFID tags,Beacons
QoS Management

Figure 1: The IoT technology layer.

White, Nallur and Clarke (2017) shows that the current QoS research initiatives largely focuses on the perception/physical and communication/network layers in the IoT architecture with minimal attention paid to the application/service layer. According to the authors, the application/service layer accounts for only 13 per cent of all the published works that examine the QoS in IoT.

This paper focuses on the QoS in the application/service layer. Given that the functionalities of the interconnected devices in IoT can be provided as a service, this layer provides the framework that allows these devices to interact with each other in a service-oriented manner (Thoma, Meyer, Spenser, Meissner and Braun, 2012). The IoT middleware, a key technology in this layer provides both functional services (e.g service abstraction,

Udoh, I. and Kotonya, G.
A Reinforcement Learning QoS Negotiation Model for IoT Middleware.
DOI: 10.5220/0009350102050212
In *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security (IoTBDS 2020)*, pages 205-212
ISBN: 978-989-758-426-8

discovery, composition and semantic interoperability) and non-functional support (e.g QoS negotiation and monitoring) (Razzaque, Milojevic-Jevric, Palade and Clarke, 2015).

QoS has been identified as a pivotal element that must be considered in IoT middleware in order to provide useful IoT services to end-user applications and actuators in critical IoT infrastructures. As IoT becomes increasingly dynamic, due to the changes in the physical world, there have been growing concerns about the best way to ensure QoS. As service consumers with varied QoS requirements (e.g hard QoS and soft QoS) interact with service providers with different QoS policy-driven behaviours (e.g energy conscious and resource-conscious), in a dynamic environment (e.g high resources and low resources), it is important to provide a mechanism that allows service providers and consumers with different QoS requirements and expectations to reach a mutually agreeable QoS resolution.

This paper presents a QoS negotiation model that uses reinforcement learning to reach a QoS agreement in an IoT dynamic environment. Considering the importance of QoS negotiation in IoT middleware, this paper presents related works in Section II, followed by the description of the negotiation environment, main assumptions and components of the QoS model. Section IV models the QoS negotiation process and section V provides details of the reinforcement learning-based QoS solution. While Section VI describes the experimental setup and the evaluation of the simulation results, section VII summarizes the paper and discusses future research directions.

2 RELATED WORKS

Negotiation is becoming an increasingly popular mechanism in maintaining QoS in IoT. Current negotiation approaches use software agents for the formation of Service Level Agreement (SLA) between service consumer and providers in IoT (Bala and Chishti, 2017). SLA-based negotiation is an important mechanism to manage the variety of requirements that characterise the IoT environment. Ghumman, Schill and Lassig (2016) designed a flip-flop negotiation strategy using the concession extrapolation and 3D utility for generation of SLA. The proposed model is useful in applications where time is a critical resource during negotiation as an agreement is reached quickly. However, this approach is characterised with a low social welfare(the sum of the utility gained by each agent) as each negotiating participant reduces their utility until the contention is resolved. Zheng, Martin, Brohman and Xu (2014) used game theory to resolve QoS contention between IoT devices and applications. This approach demonstrates a good balance between success rate and social welfare. However, it ignores the changes that may occur in the negotiating environment as it only considers the negotiating participant's action.

Alanezi and Mishra (2018) allowed negotiating parties to define their privacy requirements. while this QoS solution satisfies the privacy requirements of the negotiating participants, it is difficult for the negotiation model to be used in multi-parameter scenarios. Li and Clarke (2019) designed a QoS model that integrates a mixed negotiation strategy with the WS-Agreement Negotiation protocol for IoT service providers and consumers with intersected negotiation space. The authors considered the domain-specific properties of IoT services in their approach. However, in a multi-attribute negotiation scenario, the adopted negotiation model increases the overhead time.

Several QoS negotiation mechanisms have been proposed for web service-oriented systems such as the works of Zulkerine and Martin (2011) and Al-Aaidroos, Jailani and Mukhtar, (2011). However, due to several differences between web services and IoT services, web service negotiation methods may not be able to address the idiosyncrasies of IoT. For instance, resource-constrained and mobile devices provide IoT services, while stationary data-centres with immense computing capabilities usually host traditional web services. Thus, QoS parameters in web service systems are relatively stable and are not affected by changes that might occur in the real world, unlike the IoT ecosystem that is highly dynamic and characterised with uncertainties.

Given the dynamism of the negotiating environment of IoT, we proposed a QoS negotiation model that uses reinforcement learning and software agents to generate SLA. The idea of using reinforcement learning for QoS negotiation has been explored in the telecommunication domain. For example, Pouyllau and Carofiglio (2013) used reinforcement learning to find a QoS guaranteed path in a network that satisfies the QoS requirements and maximizes the long term goals. While their focus is on a network federation environment, our approach is directed towards the IoT middleware. Using reinforcement learning, our approach allows agents to learn how to negotiate by choosing the appropriate negotiating strategy at each stage of the negotiation process based on the changes observed in the negotiation environment and the agent's opponent offer. This approach tends to lead to a negotiation process in which an agreement is found before the deadline and offers a better balance between success rate and social welfare compared to other existing approaches.

3 QoS NEGOTIATION IN IoT MIDDLEWARE

The primary goal of this paper is to design a negotiation model that addresses the QoS contention in a dynamic environment. In order to do this, It is important we describe the environment in the middleware which the negotiation process takes place. The negotiation environment is characterized by the following:

(1) The environment contains software agents that represent IoT service providers and consumers. The service providers are IoT devices that provide sensing capabilities while the service consumers are IoT devices that provide actuation capabilities or end-user applications that provides an interface through which an IoT service can be accessed.

(2) The agents negotiate over a set of negotiable QoS parameters e.g response time and availability

(3) The environment is dynamic in nature as the QoS parameters profile, negotiation deadline and negotiation resources(e.g. CPU time and memory allocation) could change due to changes that may occur in the physical world. For example, the battery of an IoT device running low can automatically change the QoS parameters profile and an increased workload on the CPU of the IoT edge node can change the CPU time for a negotiation process.

(4) The environment is characterized by some uncertainties as agents do not have complete information about the state transitions in the environment and the preferences of other agents.

(5) Agents can only observe actions taken previously by other agents during the negotiation process and the negotiation deadline.

Having described the negotiation environment, we now focus our attention on the underlying assumptions our QoS negotiation model is based on

Assumption1 (Utility function-based offers). Agents take turns in a making offer in each round in the set {R = 0,1…$R_{deadline}$}.An offer contains an n number of negotiable QoS parameters and each QoS parameter can take a value of (q_n) within its range of permissible values ($q_n^{preferred}$ ... $q_n^{reserved}$) in the QoS profile. We adopted the microeconomics utility function described in Besanko and Braeutigam (2010) for the agents to generate offers. To model the non-linear changes associated with the QoS parameters of IoT devices, the negotiating agents use the general exponential function to map each QoS parameter value to a utility value ($U_n(q_n)$) as seen in (1):

$$U_n(q_n)= \begin{cases} \frac{e}{e-1} \times (e^{-q_n} - e^{-1}) \\ \frac{1}{e-1} \times (e^{-q_n} - 1) \end{cases} \tag{1}$$

where q_n is a real number ($0 \leq q_n \leq 1$)

With the utility value of each parameter defined, we assume that the utility of each of the QoS parameter is linearly additive. Thus the utility value of an offer can be defined as the weighted sum of the individual utility as in (2):

$$\text{U(s)} = \sum_{n=1}^{n} w_n \times U_n(q_n) \tag{2}$$

where w_n = normalized weight for each QoS parameter ($\sum_{n=1}^{n} w_n$=1)

Based on equation (2), an agent reserved offer is the sum of the utility value derived from $q_n^{reserved}$ and an agent preferred offer is the sum of the utility value derived from $q_n^{preferred}$.

Assumption 2 (Difference in Preference). We assume agents have different, sometimes opposite preferences over QoS parameters and pursue their self-interests.

Assumption 3(Dynamic and Unknown Agreement Zone). For all negotiation sessions, there exists an agreement zone which can change during the negotiation process and the agents are unaware of its location or presence.

Assumption 4 (Termination of Negotiation). The negotiation process terminates only when an agreement is found or the deadline elapses as agents are not permitted to withdraw from the negotiation process.

Now that the assumptions have been highlighted, the main components of the proposed model are described below

- *QoS parameter:* These are the non-functional attributes of an IoT service over which agents negotiate over. Agents can negotiate over more than one attributes in a negotiation session;
- *Negotiation Protocol:* This specifies the set of rules that characterise the interaction between agents (Parkin, Kuo and Brooke,

2006). It defines the negotiation states, events that can cause a change in the negotiation state and the valid actions of the agents in the different states. In our model, we adopted the stacked alternating offer protocol (Aydogan, Festen, Hindriks and Jonker, 2017). This protocol was chosen because of its support for both bilateral and multilateral negotiations as the IoT middleware is required to support both forms of negotiation;

- *Negotiation Strategy:* This is the function that enables agents to generate offers. In our model, agents can decide to choose either the concession strategy or the trade-off strategy(Zheng, Martin and Brohman, 2012). These strategies use the utility function to generate an offer. Agents are required to decide which strategy to use for each step in the negotiation process, that will maximize their utility and reach an agreement before the deadline elapses.

4 MODELLING THE QoS NEGOTIATION

The dynamism that characterizes the negotiation environment necessitated the modelling of the QoS negotiation as a Markov Decision Process (MDP) as agents are required to make decisions under these conditions. MDP presents a standard formalism to describe multistage decision making in a dynamic environment (Puterman, 2005). MDP is defined by the following elements:

- S = a set of possible states, known as the state- space, with an initial state s_0;
- A = a set of possible actions in a state s, known as the action-space;
- P = the transition model, where $P(s'|s,a)$ is the probability that action a∈A executed in state s∈S will transition to state s′∈S;
- R = the reward function, where R(s|a) is the reward function received by executing action a∈A in state s∈S;
- π = the sequence of decisions(policy) that is responsible for the mapping of states to actions. The policy can be stationary ($\pi = \pi, \pi \dots \pi$) or non-stationary($\pi = \pi_0, \pi_1, \pi_2 \dots \pi_n$).

The objective in a standard MDP is to find the optimum policy (π*) that yields the maximum sum of discounted rewards over an infinite period.

Based on these key concepts of MDP, we model the QoS negotiation as a set of n MDPs. We have n processes with each agent having its own view of the dynamics of the negotiating environment. Given the description of the negotiation environment and the assumptions made, the MDP inspired negotiation process is characterized by the following:

- *Finite deadline*: Unlike the standard MDP, the negotiation process has a finite deadline, so agents are required to maximise the expected reward over a finite timeframe;
- *Discrete state-space*: The negotiation space is defined by the availability of resources for the negotiation process, negotiation deadline and QoS profile parameters. The changes associated with the elements of the negotiation state-space results in the transition from one discrete state to another;
- *Non-stationary policy:* Since the negotiation environment is dynamic, the policy associated with the negotiation process is non-stationary. The optimum policy is determined by a reinforcement learning method;
- *Estimated transition model*: Due to the dynamics of the negotiation environment, agents do not exactly know the probability of state transition. In modelling this uncertainty in this QoS negotiation context, we defined a non-stationary estimation function, T_n to estimate the transition model and is given as:

$$T_n(s, a, s'): P_n(s, a, s') \rightarrow [0,1] \qquad (3)$$

where $n \in 0,1 \ldots\ldots N$ (nth time-step);

- *Action-space*: This defines the negotiation strategy. In our negotiation problem, agents can choose either the concession strategy or the trade-off strategy. The concession strategy enables an agent to generate offers that are of lower utility value to the offer received. Selecting the tradeoff strategy allows an agent to generate offers that are attractive to other agents while maintaining its desired utility value. This is achieved by yielding on its less preferential QoS parameters and demanding more on its more preferential QoS parameters;
- *Reward function*: Agents are rewarded based on the strategy chosen at a given state. An agent is rewarded for choosing the trade-off strategy when there is sufficient time and

resources for the negotiation process and the current range of permissible values of the QoS parameters is large. Similarly, an agent is rewarded for choosing the concession strategy if the time and resources for the negotiation process is running out and the current range of permissible values of the QoS parameters are small.

5 THE REINFORCEMENT LEARNING NEGOTIATION ALGORITHM

In this section, we begin by first introducing how we can to approximately compute the optimal policy, the optimal sequence of strategies to be adopted by the negotiating agents during the negotiation process to generate a QoS agreement. To achieve this, we adopted a model-based reinforcement learning method, *value iteration* in estimating the optimum policy (Schwartz, 2014). The value iteration method was chosen based on the fact that it is not computationally expensive and it uses less time to compute the optimal policy.

The first step in computing the optimal policy is to introduce the concept of *state value-function.* In reinforcement learning, the state value-function defines the expected cumulative reward for an agent beginning at a particular state s and under a specific policy π (Alpaydin, 2014). Formally, the state value-function at the nth time-step during the negotiation process is defined as:

$$V_n(s, \pi) = \sum_{t=n}^{N} E(R_t(s_t, \pi_t) \mid s_n = s) \quad (4)$$

where s_n is the system state at the nth time-step, R_t is the reward at the nth time-step and $E(R_t(s_t,\pi)|s_n = s)$ is the expected reward value at state $s_n = s$ and based on policy π_t. Equation (4) can be rewritten as:

$$V_n(s, \pi) = R(s,a) + \sum_{s'} P_n(s, a, s') \times V_{n+1}(s', \pi) \quad (5)$$

where $s' \in S_{n+1}$ and a is the action (strategy) determined by policy π. Amidst all the possible state value functions, it has been proven that an optimal policy π^* that has the highest value exists for any $s \in S$. Additionally, for an arbitrary V_0, the sequence of V_n has been proven to converge to V^* under the same conditions that guarantee the existence of V^* (Russell and Norvig, 2014). The state value-function of the optimal policy is given by:

$$V_n(s,\pi^*) = \max_a[R_n(s,a) + \sum_{s'} P(s, a, s') \times V_{n+1}(s',\pi^*)] \quad (6)$$

Given the uncertainty of the state-transition in the negotiation process, we adapted equation (3) to determine the expected transition model, which is given by:

$$E_n(P_n(s, a, s')) = \sum_{s'} (T_n(s, a, s') \times P_n(s, a, s')) \quad (7)$$

Combining Equation (6) and (7), we have:

$$V_n(s,\pi^*) = \max_a\{R_n(s,a) + \sum_{s'} E_n(P_n(s, a, s') \times V_{n+1}(s',\pi^*)\} \quad (8)$$

Apart from the state value-function, reinforcement learning presents another function known as the Q-function, which is a function of a state-action pair that returns a real value. The fundamental notion of the optimal Q-function is that it can be defined by the right-hand side of equation (8) as :

$$Q^*(s,a) = R_n(s,a) + \sum_{s'} E_n(P_n(s, a, s') \times V_{n+1}(s',\pi^*) \quad (9)$$

By this definition, the optimal Q-function $Q^*(s,a)$ equals the sum of the immediate reward carrying out action a in state s and the discounted expected reward after transiting to the next state s'. Thus we can obtain the relationship between $V_n(s,\pi^*)$ and $Q^*(s,a)$ and this is expressed as :

$$V_n(s,\pi^*) = \max_a[Q^*(s,a)] \quad (10)$$

Once the $Q^*(s,a)$ for all the states is known, then the optimal policy can be found. Based on this, an agent can always select an action a that maximizes $Q^*(s,a)$ in each state.

Based on the negotiation environment state-space and the agent's opponent's offer, an agent is required to either accept the opponent offer or decide which strategy(concession or trade-off) to adopt to generate an offer. An offer is accepted by an agent if the utility value of the received offer is far greater than its utility value of its reserved offer. The algorithmic descriptions of both the concession and trade-off strategy are presented in (Zheng et al., 2012). These strategies were chosen in our proposed algorithm because we can control the rate at which they converge.

Having specified the negotiation strategies for generating offers and reinforcement method for selecting at a strategy at a specific time-step, we now introduce our reinforcement learning negotiation algorithm as shown in Algorithm1. This algorithm will enable negotiating agents to appropriately map a strategy to a negotiation state resulting in the timely discovery of a QoS solution. The algorithm begins by allowing an agent to observe the negotiating participant's offer which is usually an offer with a high utility for the agent's opponent. Given that the condition in line 2 is true, it proceeds to iteratively compute $V_n(s,\pi^*)$ for each time-step and all the states

Algorithm 1: Reinforcement Learning Negotiation Algorithm.

Input : - *The negotiating opponent offer (Y_i)*
- *The deadline criterion*
- *Array B with the best and worst values for n QoS parameters*
- *Array C with the weights of n QoS parameter*
- *Array D with flags of n QoS parameter; A flag indicates if a QoS parameter preferred value is greater than its reserved value.*
- *The reward function $R_n(s,a)$ for each (s,a)*
- *Parameter λ_1 and λ_2 ($0 < \lambda_1, \lambda_2$), indicating the degree of concession and trade-off at instant n respectively*
- *The estimation function for the state transition, $E_n(P_n(s, a, s'))$ at instant n*

Output: true if it is a success, otherwise false.

```
1.  Offer $Y_i$ is presented
2.  while $Y_i$ is not accepted
3.  Compute iteratively
    $V_n(s,\pi^*)=\max_a\{R_n(s,a)+\sum_{s'} E_n(P_n(s,a,s') \times V_{n+1}(s',\pi^*)\}$
4.  select the action that
     maximises $V_n(s,\pi^*)$
5.  if a=concession then
6.      $k_1 \leftarrow k_1+1$
7.      $Y_{i+1} \leftarrow$ concession(A,B,C,$k_1$,$\lambda_1$)
8.  else
9.      $k_2 \leftarrow k_2+1$
10.     $Y_{i+1} \leftarrow$ trade-off (A,B,C,$k_1$,$\lambda_2$)
11. k $\leftarrow k_1+k_2$
12. if $Y_{i+1}$ is out of bounds or
    deadline criterion is reached
    then
13.    return FALSE
14. else
15.    offer $Y_{i+1}$ is presented
16. return TRUE
```

including the terminal states((i.e states. indicating that the deadline is reached, an agreement is found or the negotiating resources in the IoT middleware are exhausted). With the present state of the system determined, it selects the action(i.e concession strategy or trade-off strategy) that maximizes the optimum Q function. The selected strategy is used in generating a counter-offer. If the generated offer is beyond the defined bounds or the deadline criterion is reached, *FALSE* is returned; otherwise, *TRUE* is returned and the counter-offer is presented. A QoS solution is found when the condition in line 2 is true. The reinforcement learning paradigm described in Russel et al. (2014) and Zheng, et al. (2012) provides proof that the reinforcement learning negotiation algorithm converges and terminates after a finite number of time-steps.

6 EVALUATION

To evaluate our algorithm, we considered a simple bilateral negotiation scenario involving two negotiation agents where the first agent represents the IoT service provider and the other agent represents the IoT service consumer. These agents were implemented using the open-source project, Java Agent DEvelopment Framework (JADE) (Bellifemine, Giovanni and Dominic, 2007). We implemented the reinforcement learning negotiation algorithm using Java and all the experiments were conducted on a Lenovo laptop with a 2.50GHz Intel i5 processor with 8GB memory running Windows 10. The concept of multi-threading in Java was used to model the behaviour of both agents as we utilize the thread synchronization method to model the exchange of offers and counter-offers. We developed a QoS profile in a JSON format based on the ideas in the IoT literature of Villalonga, Bauer, Aguilar, Huang and Strohbach (2010), which defines its service and three QoS parameters: response-time (RT), availability(AVAL) and throughput(TP). The agents preference over these QoS parameters are kept private and the negotiation process begins with the IoT service consumer presenting its preferred offer. We compared the reinforcement learning negotiation model with the mixed strategy negotiation model described in Zheng et al. (2014) that uses a random probabilistic model for the selection of a strategy during the negotiation process. This strategy was chosen for the comparison because it inspired our algorithm and it provides a good balance between the success rate and social welfare. We used three metrics for the comparison: negotiation turns, success rate and social welfare.

To simulate the dynamics in the negotiation environment, we had 8 states(2 negotiation deadlines[D^1, D^2], 2 states of resource availability [low, high] and 2 utility values for the reserved [U^1, U^2]) and 64 state transitions. Each of the state transitions is associated with a strategy, estimation function and reward function. Table 1 below shows a snippet of this association. We conducted a series of experiment with different permissible values and weights for each of the QoS parameters of each agent. In all the experiments, the values of λ_1 and λ_2 were fixed at 0.10. While the difference between the preferred value of both agents for the RT, AVAL and TP was set at a maximum 0.45, 0.60 and 0.55, that of the reserved value was set to a maximum of 0.30,0.23, 0.15 respectively. Similarly, the difference in the deadline criterion was set at 15. Table 2 shows an example of a specific set of values for the negotiation parameters.

Table 1: The State Transitions Table.

Current State (s)	Next State (s′)	Action (strategy)	EF	RD
[low, U^1,D^1]	[low, U^1,D^1]	Concession	P^{CS}	R^{CS}
[low, U^1,D^1]	[high, U^1,D^1]	Concession	P^{CS}	R^{CS}
[low, U^1,D^2]	[high, U^1,D^2]	Trade-off	P^{TO}	R^{TO}
[low, U^1,D^2]	[high, U^2,D^2]	Trade-off	P^{TO}	R^{TO}

EF→ Estimation Function $P_n(s, a, s')$
RD→Reward Definition $R(s,s'|a)$

Table 2: Sample Negotiation Parameters.

		IoT Service Consumer	IoT Service Provider
RT	weight	0.3	0.4
	range	~0.92 - 0.76*	0.95* - 0.74~
AVAIL	weight	0.2	0.4
	range	~0.95 - 0.81*	0.93* - 0.81~
TP	weight	0.5	0.2
	range	~0.90 - 0.83*	0.91* - 0.76~
Deadline criterion		35 rounds	50 rounds

* indicates the preferred value and ~ indicates the reserved value

In these experiments, we observed that the reinforcement learning model consistently had a better performance compared to the mixed strategy negotiation model. The average round taken to discover an agreement using our algorithm was 38% less than average rounds of the mixed strategy negotiation algorithm. In terms of social welfare, there was an average increase of 19% in the utility gained with our algorithm. These improvements in performance are primarily based on the selection of a reward scheme that maintained a balance between reaching an agreement before the deadline and a high social welfare thereby making the agents select the appropriate strategy based on the changes that occur in the negotiation environment. To demonstrate this, Figure 2 and Figure 3 illustrates the results of some selected negotiation sessions comparing both approaches in terms of social welfare and negotiation turns. Since the reinforcement learning model and the mixed strategy model made use of algorithms that have been proven to converge at a finite period, they achieved roughly the same average success rate of 98.74% and 95.12% respectively.

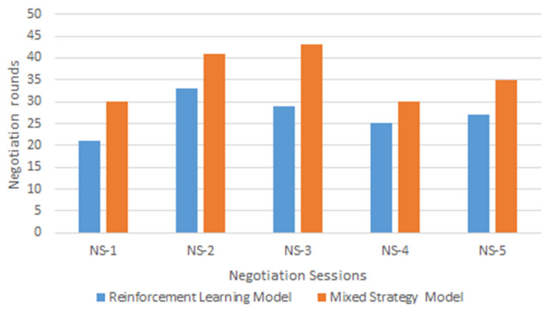

Figure 2: Negotiation rounds results for reinforcement learning model and mixed strategy model.

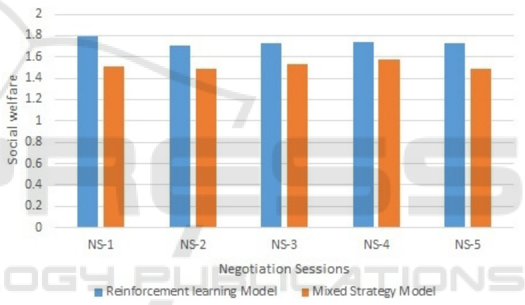

Figure 3: Social welfare results for reinforcement learning model and mixed strategy model.

7 CONCLUSION

This paper introduces a model-based reinforcement learning algorithm for the QoS negotiation in the IoT ecosystem. The algorithm selects the best strategy to be used by negotiating agents at each time-step during the negotiation process when the IoT environmental dynamics are probabilistically known. This selection is based on the observed changes in the negotiation environment. Preliminary results demonstrate that our approach of using reinforcement learning tends to lead to a negotiation process in which an agreement is found before the deadline and offers a better success rate and social welfare compared to the approach that uses a random a probabilistic algorithm.

Although in this paper we restricted our negotiation model to bilateral negotiation scenario, it can be easily extended to multilateral negotiation scenario given that the negotiation protocol adopted supports multilateral negotiation. In the future, we plan to extend the QoS model that takes into account situations where system dynamics is not probabilistically known.

REFERENCES

Patel, K. K., and Patel, S. M. (2016). Internet of Things - IoT: Definition, characteristics, architecture, enabling technologies, application & future challenges. *International Journal of Engineering Science and Computing*, 6(5), 6122-6131.

Duan, R., Chen, X., and Xing, T. (2011). A QoS architecture for IoT. *2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing*, 717-720.

White, G., Nallur, V., and Clarke, S. (2017). Quality of service approaches in IoT: A systematic mapping. *Journal of Systems and Software*, 186-203.

Thoma, M., Meyer, S., Sperner, K., Meissner, S., and Braun, T. (2012) On IoT-services: Survey, Classification and Enterprise Integration. *2012 IEEE International Conference on Green Computing and Communications*, 257-260.

Razzaque, M. A., Milojevic-Jevric, M., Palade, A., and Clarke, S. (2015) Middleware for Internet of Things: A Survey. *In IEEE Internet of Things Journal*, 3(1), 70-95.

Issarny, V., Bouloukakis, G., Georgantas, N., and Billet, B. (2016). Revisiting Service-Oriented Architecture for the IoT: A Middleware Perspective. *International Conference on Service-Oriented Computing*,1-16.

Bala, M. I., and Chishti, M. A.(2017) A model to incorporate automated negotiation in IoT. *2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 1-4.

Ghumman, W. A., Schill, A., and Lässig, J. (2016). The Flip-Flop SLA Negotiation Strategy Using Concession Extrapolation and 3D Utility Function. *2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC)*, 159-168.

Zheng, X., Martin, P., Brohman, K., and Da Xu, L. (2014). Cloud Service Negotiation in Internet of Things Environment: A Mixed Approach. *In IEEE Transactions on Industrial Informatics*, 10(2), 1506-1515.

Alanezi, K. and Mishra, S. (2018). A privacy negotiation mechanism for IoT. *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/ CyberSciTech)*, 512-519.

Li, F. and Clarke, S. (2019). A Context-Based Strategy for SLA Negotiation in the IoT Environment. *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)* 208-213.

Zulkernine, F. H. and Martin, P. (2011). An adaptive and intelligent SLA negotiation system for web services. *IEEE Transactions on Services Computing,* 4(1), 31–43.

Al-Aaidroos, M., Jailani, N. and Mukhtar, M. (2011). Agent-based negotiation framework for web services SLA. *2011 7th International Conference on Information Technology in Asia*, 1-7.

Pouyllau, H., & Carofiglio, G. (2013). Inter-carrier SLA negotiation using Q-learning. *Telecommunication Systems*, *52*(2), 611-622.

Besanko, D. and. Braeutigam, R. R.(2010) *Microeconomics,* Wiley. New Jersey, USA, 4th edition.

Parkin, M., Kuo, D., and Brooke, J. (2006). A Framework and Negotiation Protocol for Service Contracts. *2006 IEEE International Conference on Services Computing (SCC'06)*, 253-256.

Aydoğan, R., Festen, D., Hindriks, K. V., and Jonker C. M. (2017). Alternating Offers Protocols for Multilateral Negotiation. *In Modern Approaches to Agent-based Complex Automated Negotiation. Studies in Computational Intelligence,* Springer, Switzerland, 153-167.

Zheng, X., Martin, P., and Brohman, K. (2012). Cloud service negotiation: Concession vs. tradeoff approaches. *In Proc. 12th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGrid 2012)*. pp. 515–522.

Puterman, M. L. (2005). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, Wiley & Sons. New Jersey, USA.

Schwartz, H. M. (2014). *Multi-Agent Machine Learning: A Reinforcement Approach*, Wiley & Sons. New Jersey, USA.

Alpaydin, E. (2014). *Introduction to Machine Learning,* MIT Press. Cambridge, MA, USA, 3rd edition.

Russell S., and Norvig, P. (2014). *Artificial Intelligence: A Modern Approach,* Pearson. Essex, England, 3rd edition

Bellifemine, F. L., Caire, G., and Greenwood, D. (2007). *Developing Multi-Agent Systems with JADE,* Wiley & Sons. England.

Villalonga, C., Bauer, M., Aguilar, F. L., Huang, V. A., and Strohbach, M. (2010). A resource model for the real world internet. *In European Conference on Smart Sensing and Context*. 163–176.