Intelligent Service Fulfillment for Software Defined Networks in Smart City

Md. Shirajum Munir, Sarder Fakhrul Abedin, Md. Golam Rabiul Alam,
Nguyen H. Tran, Choong Seon Hong
Department of Computer Science and Engineering
Kyung Hee University
Suwon, South Korea.
{munir, saab0015, robi, nguyenth, cshong}@khu.ac.kr

***Abstract*—Smart city is the prospect of current intellectual technology toward the ecological growth of urban technology and commercial expansion. In addition, the Internet of Things (IoT) based smart city services ensure the eminence of life and well-being to the smart citizens. In order to ensure quality services, each of city service gathers multidimensional data from abundant IoT nodes. Therefore, the centralized traffic management has become critically challenging for a large volume of multidimensional smart city network data. Consequently, in this research, we concentrate on solving this problem by introducing *Software Defined Networks (SDN) based intelligent service fulfillment* for dense smart city network to accomplish service requests from multiple smart citizens and service providers. First, we model a distributed software defined IoT network for smart city environment where introduce a fog-based SDN controller with an intelligent engine, fog unit, and virtual mesh topology module. Then, we propose a reinforcement learning (RL) based intelligent algorithm for IoT network, and that intelligent agent-based service fulfillment algorithm accomplishes the city service. We use fog based SDN controller unit to implement the learning model based on processing and waiting delay of the SDN switches that qualify the smart city services to the service providers. Finally, in the simulation, we have achieved higher performance gain for the proposed method in respect to the convergence and utility gain.**

***Index Terms*—smart city service, distributed network, intelligent agent, software defined networking, reinforcement learning**

I. Introduction

The smart city shall be intelligent to incorporate reasonably and effortlessly with a massive number of heterogeneous network infrastructure with microcontrollers, sensors, and actuators based devices. Based on those millions of devices, the smart city consists of thousands of city services for monitoring and smart service facilitation to the citizens. The data-centric services such as energy, water, security, gas-supply are not only useful but also sensitive for the effective decision making in the smart city environment. Moreover, a large amount of multidimensional and dense data are being generated by the sensors that are deployed for the data-centric services. To achieve a higher degree quality of services, it needs artificial intelligence for real-time decision making.

A. Background and Motivations

IoT architectures, protocols, smart city data aggregation, along with an extensive variety of facilitating technologies obtainable researches are ongoing for the IoT based Smart city networks [1] [2] [3].

However, researchers are introducing artificial intelligence in smart city network. Semi-supervised deep reinforcement learning based infrastructure suggested for smart city indoor and outdoor services to improve the QoS for smart services [4].

Consequently, fog computing technology used for low range device to device communication for the IoT network [5]. To enhance the fog computing for the different smart city services a service-oriented middleware (SOM) has proposed. [6].

Nowadays, in the smart network scenario, SDN based solution is considered to be more reliable and accessible in terms of execution time and network manageability. More specifically, the IoT based smart infrastructures are likely to be managed by the SDN and virtual solution. For example, the industrial IoT (IIoT) like smart grid is provided with the SDN platform to support resiliency in case of the real-time monitoring of smart grid networks [7]. Moreover, for intelligent transport systems, SDN is introduced as a prominent solution for enabling better QoS, network management, and real-time decision making [8].

Hence, the distributed SDN network control system is more important for network management and decision making where a large portion of multidimensional sensors data are generated from the congested smart city network [9]. An adaptive consistency model for SDN controller is introduced for distributed control plane to ensure the switch synchronization within a distributed manner [10]. Therefore, some groups of researchers worked with the ubiquitous flow control and mobility management that used distributed controllers and distributed hashing based overlay structure for software-defined IoT system [11]. Additionally, distributed urban sensing is one of the major contributions to software-defined networking for IoT [12].

However, to solve data aggregation scheduling problem, distributed delay efficient data aggregation scheduling is being introduced for duty-cycled wireless sensor networks [13].

B. Intelligent SDN in Smart City

In this research, we focus on intelligent agent-based smart city service fulfillment for distributed SDN network. Service fulfillment is a common challenge in the dense smart city network. To minimize the smart city service time, we consider a fog-based intelligent SDN controller into distributed SDN in a smart city. In the smart city network, each service provider can make request for a service fulfillment through the intelligent agent. Additionally, these requests are served by the fog based SDN controller.

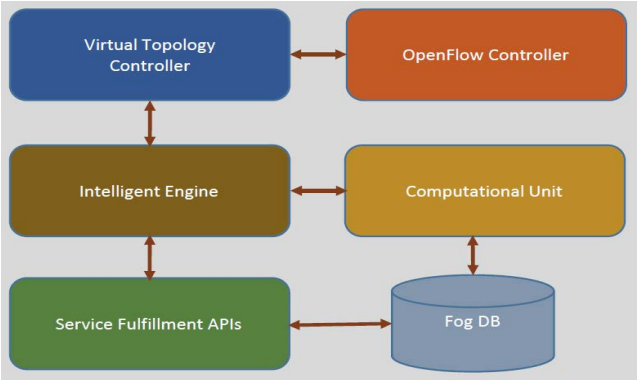

Fig. 1. Fog based SDN controller architecture.

The proposed fog-based SDN controller architecture is shown in Fig 1. Fog computing reduces the network traffic communication cost by using its storages and computation capability. [14]. In our proposal, we include the SDN controller into the fog AP. In this architecture, a virtual topology unit maintains the mesh topology of the networks. We use M/M/1 queuing model for SDN switches [15]. In the intelligent engine, we apply the reinforcement learning [16] module based on the service delay, payload traveling delay and waiting delay of the SDN switches. Consequently, for maximizing the switch throughput, intelligent engine periodically executes the learning algorithm. Additionally, the computational unit and fog storages units reduce the network overhead for data aggregation and decision making compared to the cloud computing, these ensure the seamless services fulfillment into a smart city.

In the proposed fog based SDN controller, the service providers can make requests through the service fulfillment APIs. Therefore, these APIs communicate through the intelligent engine for service fulfillment. Also, the intelligent engine ensures the decision making for serving those request with the help of computational unit. However, computational unit is capable to perform the data processing for service requests. Finally, service fulfillment APIs provide the response to the service provider with service data. Intuitively these service fulfillment APIs are the intelligent agent.

Under the above circumstances, the main contributions of this paper are,

- We model a distributed smart city software-defined network that includes a fog-based SDN controller. This fog based SDN controller maximize the performance of a large volume multidimensional data aggregation to fulfill the smart city service requests for the service providers and smart citizens. And also the intelligent engine unit ensures the maximum utilization of SDN switch capacity by using the artificial intelligence. Therefore, the distributed network minimize the risk of network outage.
- We formulate a delay sensitive service fulfillment problem for smart city services, and we show that the service fulfillment is a NP-hard problem. To solve this problem, we use model-based reinforcement learning to fulfill the city services. Furthermore, the intelligent agent-based algorithm minimizes the overall queuing, transmission, and processing delay to improve the service fulfillment QoS.

This paper is organized as follows, section II presents the system model and problem formulation of intelligent agent service fulfillment problem. Section III provides the solution approach by using model-based reinforcement learning and algorithm design. Simulation and performance analysis are included in Section IV. Finally, Section V provides the conclusion of the proposed research.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Model Description

The proposed system model of SDN based intelligent service fulfillment for distributed smart city IoT network is presented in Fig 2. We have proposed a distributed SDN network for a smart city where SDN local controller also works as a fog node and each controller is connected with mesh topology. SDN switches and APs are also in a mesh topology. Local SDN controller can control a particular part of the network. All the local controller connected to the base station and in the base station, there is a global SDN controller. The role of the global controller is to manage the local controllers and facilitate data uploading to the cloud server. Here, we also introduce intelligent module for both local and global controller where we apply artificial intelligence to maintain the network scalability, consistency and reliability action for smart city service requests. The APs and IoT nodes are equipped with wireless connectivity. whereas the BTS and core network are connected with backhaul connection. In addition, the Cloud is also connected to the core network. The Intelligent agent is a part of the fog based SDN local controller. There are multiple agents in the network. Every switch maintains the queuing model for serving the requests.

We consider i number of SDN local controllers $\mathcal{C} = \{C_1, C_2, ..., C_m\}$ and j number of SDN switches $\mathcal{S} = \{S_1, S_2, ..., S_n\}$ in the smart city network. There are many service provider in the network. For describing the scenario, we are now considering the video surveillance system as a service. In this scenario, the security manager needs to perform security surveillance for a particular Geo-area. There are some static security cameras installed for real-time sensing operation to gather vital security information.

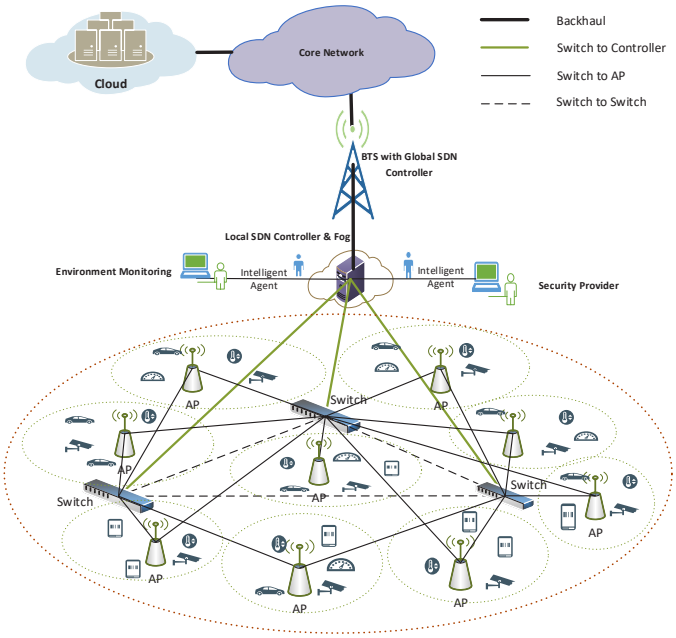

Fig. 2. System model.

We assume that SDN switches and APs are connected with mesh topology along with the SDN local controller. The SDN local controller is responsible for managing the traffic flow for its associated switches. All switches of the SDN in smart city network maintains M/M/1 queuing model for service fulfillment. In this scenario, we guarantee the QoS of smart city service fulfillment by improving the switch selections model. Those switch selections are controlled by the local controller that can be able to select a particular AP based on minimum processing time and queuing delay for payload transfer. The local SDN fog controller aggregates the data and sends it to the service provider that minimizes the uplink and downlink delay between a service request and cloud server and vice-versa. Here the central SDN controller periodically uploads the necessary payload to the cloud.

B. Intelligent Service Fulfillment Problem for SDN in Smart

TABLE I
SUMMARY OF NOTATION

Notation	Description
$\mathcal{C}$	Set of the local controllers $\mathcal{C} = \{C_1, C_2, ..., C_m\}$
$\mathcal{S}$	Set of the switches $\mathcal{S} = \{S_1, S_2, ..., S_n\}$
$t_{i,j}$	Traverse Delay from C_i to S_j
p_j	Processing delay in S_j
λ_i	Arrival rate from C_i
$\mu_{i,j}$	Service rate from C_i to S_j
α_i	Data volume transfer rate
T_i	Total travel time from C_i to S_j per unit time
E_i	Total waiting time from C_i to S_j per unit time
l	Learning rate
γ	Discount rate

When a service request arrives at local fog controller C_i then, the controller sends data request with request rate λ_i to the connected switches. Switch S_j can transfer data with average data volume transfer rate α_i to local fog controller with the service rate $\mu_{i,j}$.

To ensure the service, this problem is a service fulfillment problem and it is a NP-hard problem. We can formulate the problem as an integer linear optimization programming.

$$X_{i,j} = \begin{cases} 1, & \text{if } C_i \text{ goes to } S_j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Where, $\forall C_i \in \mathcal{C}, \forall S_i \in \mathcal{S}$

$X_{i,j}$ is a binary indicator variable that gives either 0 or 1. If the controller C_i requests to switch S_j and travels through the switch i to j then it gives 1 otherwise it is 0.

Service rate from controller C_i to switch S_j is, $\mu_{i,j} = \frac{1}{\alpha_i p_j}$, where p_j is the processing delay and α_i is the data volume transfer rate at switch S_j. Now the expected waiting delay is as follow [15].

$$E_{i,j} = \frac{1}{\mu_{i,j} - \lambda_i}. \tag{2}$$

The total waiting delay in per unit time for controller C_1 is,

$$W_i = \sum_{i:\, C_i \in \mathcal{C}} \sum_{j:\, S_j \in \mathcal{S}} \lambda_i x_{i,j} E_{i,j} \tag{3}$$

Additionally, the aggregated per unit traverse time requests from the C_1 is,

$$T_i = \sum_{i:\, C_i \in \mathcal{C}} \sum_{j:\, S_j \in \mathcal{S}} \lambda_i x_{i,j} t_{i,j} \tag{4}$$

Now the delay between controller C_i and switch S_j is $D_{i,j} = \lambda_i (t_{i,j} + E_{i,j})$.

So, the accumulated delay for waiting, servicing and traversing is,

$$D = \sum_{j:\, S_i \in \mathcal{S}} (W_i + T_i) = \sum_{i:\, C_i \in \mathcal{C}} \sum_{j:\, S_j \in \mathcal{S}} x_{i,j} D_{i,j} \tag{5}$$

Now we formulate an optimization problem where the objective is to minimize the total delay that means, maximize the total utility. The objective function of utility maximizing problem represent as follow,

$$\chi : max \sum_{i:\, C_i \in \mathcal{C}} \sum_{j: S_j \in \mathcal{S}} \frac{1}{D_{i,j}} x_{i,j} \tag{6}$$

Subject to:

$$0 \le x_{i,j} \le 1 \qquad i = 1, ..., m, j = 1, ...n \tag{7}$$

$$\sum_{i=1, i \ne j} x_{i,j} = 1 \qquad j = 1, ..., n \tag{8}$$

$$\sum_{j=1, j \ne i} x_{i,j} = 1 \qquad i = 1, ..., m \tag{9}$$

$$\sum_{i:\ C_i \in \mathcal{C}} \sum_{j:\ S_j \in \mathcal{S}} x_{i,j} = 1 \quad \forall C_i \in \mathcal{C} \tag{10}$$

$$\sum_{i:C_i \in \mathcal{C}} \sum_{j:\ S_j \in \mathcal{S}} \frac{\lambda_i}{\mu_{i,j}} x_{i,j} \leq 1 \quad \forall C_i \in \mathcal{C} \tag{11}$$

$$x_{ij} \in \{0,1\}, \quad \forall C_i \in \mathcal{C}, \forall S_j \in \mathcal{S} \tag{12}$$

Constraints of this optimization problem are described by equation (7) to (12). The first constraint ensures that controller C_i send a request to switch S_j. Therefore, the boundary for the C_i to S_j controlled by the equation (8) and (9). Equation (10) guarantees that the request from the controller has already been sent. Equation (11) ensures that the arrival rate does not cross the service rate at controller C_i for switch S_j. Finally, the equation (12) is responsible for the binary decision making of $x_{i,j}$.However, the problem in (6) is an NP-hard problem.

III. REINFORCEMENT LEARNING BASED SERVICE FULFILLMENT

We have discussed the system model and problem formulation for intelligent agent-based service fulfillment for smart city environment in section II. To solve this problem we have introduced reinforcement learning based solution approach for smart city distributed SDN network.

For RL solution approach is used finite Markov Decision Process (MDP). Here, a set of the state is φ, set of action is $\mathcal{A}$ and the immediate reward function is r_j. The set of state is defined as, $\varphi = \{S_1, S_2, ..., S_j\}$.

Therefore, every switch of a local fog controller is the element of state set. So that, the controller can traverse any of them for the service request. The action set is a visit to the next switch. If it has been visited then reward will be calculated based on utility of the objective function χ and the reward of an individual switch S_j at controller C_i is,

$$r_j = \sum_{i:\ C_i \in \mathcal{C}} \sum_{j:S_j \in \mathcal{S}} \frac{1}{D_{i,j}} x_{i,j} - \tau_{i,j}. \tag{13}$$

In equation (13) $\tau_{i,j}$ is the penalty for switch S_j that reflects on equation (11). Hence, the arrival rate at controller C_i is greater than the service rate then penalty is as follow,

$$\tau_{i,j} = \begin{cases} \frac{D_{i,j}}{D_{min}}, & \text{if } \frac{\lambda_i}{\mu_{i,j}} x_{i,j} \geq 1 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

In equation (14), D_{min} is threshold value for tolerable delay.

In our learning model, to find an optimal policy in each state, we approximate the Q-value $\bar{Q}(\varphi_j, A_j)$.

$$\bar{Q}(\varphi_j, A_j) = R(\varphi_j, A_j) + \gamma * Max[Q(\varphi_{j+1} \forall A_{j+1})] \tag{15}$$

In equation (15) γ is the discount rate and $R(\varphi_j, A_j)$ is the current reward. $Max[Q(\varphi_{j+1} \forall A_{j+1})]$ means, it uses all possible actions in a state to calculate the optimal approximates. We defined the Q-function as follow:

$$Q(\varphi_j, A_j) = Q(\varphi_j, A_j) + l * \bar{Q}(\varphi_j, A_j) \tag{16}$$

In equation (16) l is the learning rate.

$$Q(\varphi_j, A_j) = Q(\varphi_j, A_j) + l * (R(\varphi_j, A_j) + \gamma * Max[Q(\varphi_{j+1}, \forall A_{j+1})]) \tag{17}$$

Equation (17) is generated by using equation (15) and (16) that is the simplified representation of the learning model.

In our solution approach, we have proposed modeled based ϵ - greedy algorithm for balancing between exploration and exploitation. Additionally, ϵ greedy algorithm converges to the optimal solution by using episodic training.

$$A_j = \begin{cases} \bar{A}_j, & \text{with probability } 1 - \epsilon \\ \Psi, & \text{with probability } \epsilon \end{cases} \tag{18}$$

Here, $\bar{A}_j$ is the selected action with probability $1 - \epsilon$ and Ψ is the random selection of action from the action set $\mathcal{A}$ with ϵ probability.

Algorithm 1: Intelligent Service Fulfillment

Input: reqItems[]
Result: ServiceData
1 **Q value Calculation:** $\sum_{i:\ C_i \in \mathcal{C}} \sum_{j:S_j \in \mathcal{S}} Q(\varphi_j, A_j)$
2 **while** $(0 \leq x_{i,j} \leq 1\ and\ \sum_{i=1, i \neq j} x_{i,j}\ and \sum_{j=1, j \neq i} x_{i,j} = 1)$ **do**
3 **foreach** $\sum_{i:\ C_i \in \mathcal{C}} \sum_{j:\ S_j \in \mathcal{S}} x_{i,j} = 1 \quad \forall C_i \in \mathcal{C}$ **do**
4 **while** $A_j = \begin{cases} \bar{A}_j, & \textit{with probability } 1 - \epsilon \\ \Psi, & \textit{with probability } \epsilon \end{cases}$ **do**
5 $r_j = \sum_{i:\ C_i \in \mathcal{C}} \sum_{j:S_j \in \mathcal{S}} \frac{1}{D_{i,j}} x_{i,j} - \tau_{i,j}.$
6 $Q(\varphi_j, A_j) = Q(\varphi_j, A_j) + l * (R(\varphi_j, A_j) + \gamma * Max[Q(\varphi_{j+1}, \forall A_{j+1})])$
7 **while** $reqItems[]$ **do**
8 ServiceData = fullFillRequest(Q, reqItems)

In Algorithm 1 we only focus on high-level steps that cover all the constraints, and functionality. In step 2, we consider first three constraints that ensure the controller C_i must visit all the connected switches, and fulfill the boundary conditions. On the other hand, step 3 ensures that utility calculation for visited switches. However, for the reward calculation at step 5, the last two constraints equation (11), and equation (12) are executed.

The policy based Q-learning average complexity is $O(en)$, and the worst case complexity is $O(en^2)$. Here, n is the all input size and e is the number of episodes. Finally, by using Q-table, service request completes with $log(n)$ time complexity. In a summary, the proposed Q-learning based model solved the NP-hard problem within a polynomial time that also ensures more robust, and scalable solution.

IV. PERFORMANCE EVALUATION

For testbed setup, we have used floodlight openflow 1.3 controllers, and Open vSwitch. Here, we used mininet emulator module for switches, and host (APs) implementation.

We have considered every AP as host and Floodlight open-flow controller web GUI have been used for performance monitoring. Finally, we have used python 3.6 platform for ϵ-greedy algorithm based reinforcement learning algorithm implementation, and performance evaluation. Fig. 3 shows the mininet emulating environment topology.

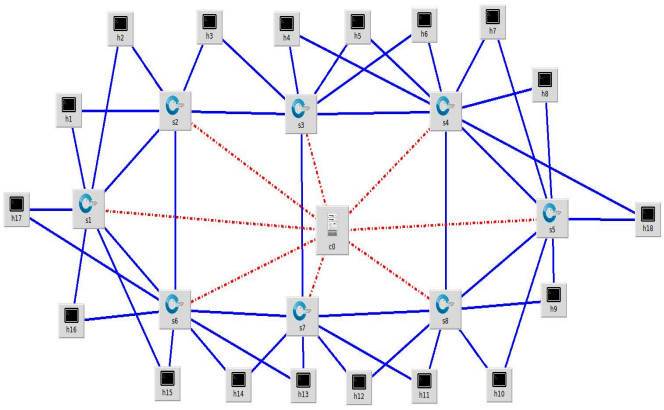

Fig. 3. Mininet topology for a single controller.

Here one fog SDN controller consists 8 switches, and 40 individual APs for simulation. Service requests are generated through the service provider, and for each service, we have used different sizes from 64KB to 128MB of payloads. Fig. 4 is the floodlight controller GUI topology view at runtime.

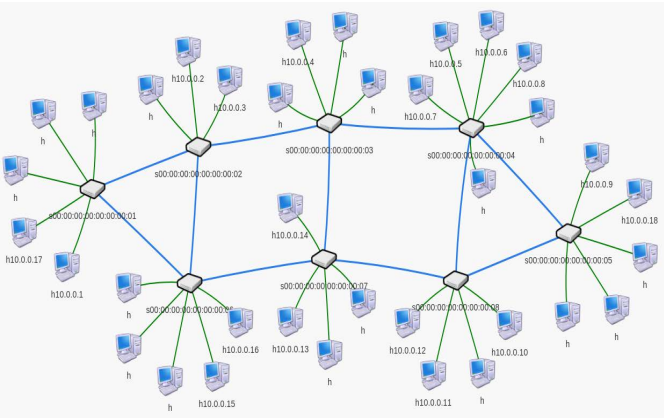

Fig. 4. Executing topology from floodlight controller GUI.

In Fig. 5 shows the number of episode vs. score value for different ϵ values 0.10, 0.15, and 0.25. Here, after the 120 episode, it has reached to convergence and is it an optimal solution.

Fig. 6 describes the convergence statistics for ϵ value at each episode. It is noticeable that after the 120 episode, it reaches to convergence for all different starting ϵ value.

As our problem is a delay minimization problem, it maximizes the utility value of reward calculation. Fig. 7 shows the waiting time vs. utility analysis. Here, the utility is the inverse of normalized delay for the service task. When the total

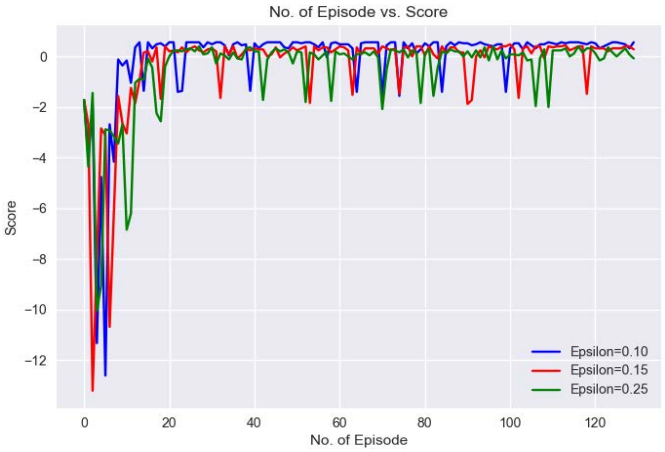

Fig. 5. Score value vs. no. of episode.

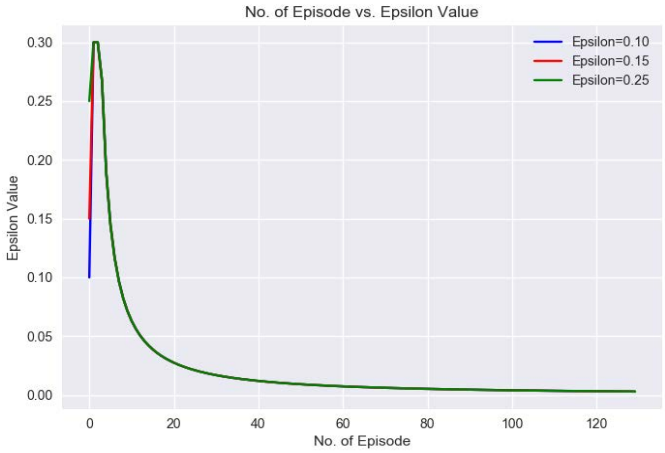

Fig. 6. Convergence result.

delay is 0.10 ms then, it provides the maximum utility. That means it maximizes the optimal usage of resources. On the other hand, when the delay is 0.20 ms, the resource utilization is insignificant for that task. It is clear that if there is less processing delay then, it improves the QoS.

The average time complexity of our intelligent agent fulfillment solution is $O(n^2)$ and the optimal solution for TSP is around $O(2^n n^2)$ where the complexity of each subproblem is $O(2^n n)$. The round-trip time comparison described in Fig. 8. This figure describes the service fulfillment execution time RTT (round-trip time) for 1 to 30 service request simultaneously through the intelligent agent. The dot line (red) is based on average case time complexity and rectangle line (green) depicts the time complexity of the proposed intelligent agent-based service fulfillment algorithm. For multiple services, our proposed method shows significant performance gain.

In a summary, from the simulation results we observe that, our proposed intelligent service fulfillment model for SDN in smart city network has achieved a higher degree of performance gain compare to others.

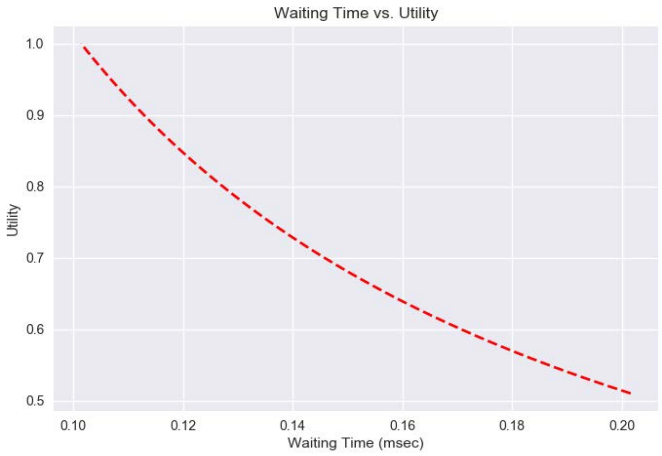

Fig. 7. Waiting time vs. utility.

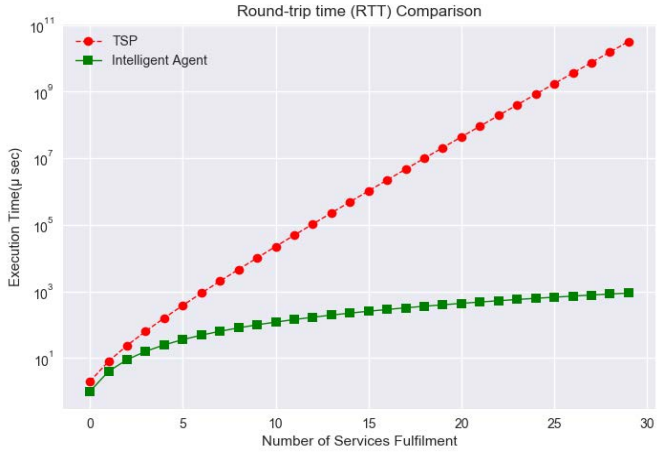

Fig. 8. Service fulfillment RTT.

V. CONCLUSION

SDN based intelligent service fulfillment is a novel approach that enables an extensive variety of smart city services. Furthermore, the proposed method comprehensively reduces the network overhead by introducing the distributed network management and intelligent engine based delay sensitive service fulfillment technique. Additionally, the fog computational unit and virtual mesh topology significantly improve the real-time data aggregation where the implementation of distributed switches are considered instead of a single switch. Our proposed model is also capable of reducing the risk of network failure in the dynamic environment like smart city.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (NRF-2017R1A2A2A05000995).

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the Grand Information Technology Research Center support program (IITP-2017-2015-0-00742) supervised by the IITP(Institute for Information & communications Technology Promotion)" *Dr. CS Hong is the corresponding author.

REFERENCES

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for smart cities," IEEE Internet Things J., vol. 1, no. 1, pp. 2232, Feb. 2014.

[2] S. F. Abedin, M. G. R. Alam, R. Haw and C. S. Hong, "A system model for energy efficient green-IoT network,"2015 International Conference on Information Networking (ICOIN), Cambodia, pp. 177-182, 2015.

[3] M. S. Munir, M. G. R. Alam, C S Hong, "Smart Agent Based Data Aggregation for Smart City", 2017 Korea Computer Congress (KCC 2017), June. 2017.

[4] M. Mohammadi, A. Al-Fuqaha, M. Guizani and J. S. Oh, "Semi-supervised Deep Reinforcement Learning in Support of IoT and Smart City Services," IEEE Internet of Things Journal, vol. PP, no. 99, pp. 1-1, 09 June 2017.

[5] S. F. Abedin, M. G. R. Alam, N. H. Tran, C. S. Hong, "A Fog based System Model for Cooperative IoT Node Pairing using Matching Theory," The 17th Asia-Pacific Network Operations and Management Symposium(APNOMS 2015), Aug 19-21, 2015.

[6] N. Mohamed, J. Al-Jaroodi, I. Jawhar, S. Lazarova-Molnar and S. Mahmoud, "SmartCityWare: A Service-Oriented Middleware for Cloud and Fog Enabled Smart City Services," IEEE Access, vol. 5, no. , pp. 17576-17588, 2017.

[7] S. Al-Rubaye; E. Kadhum; Q. Ni; A. Anpalagan, "Industrial Internet of Things Driven by SDN Platform for Smart Grid Resiliency," IEEE Internet of Things Journal , vol.PP, no.99, pp.1-1, August 2017.

[8] X. Wang, C. Wang, J. Zhang, M. Zhou and C. Jiang, "Improved Rule Installation for Real-Time Query Service in Software-Defined Internet of Vehicles," IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 2, pp. 225-235, Feb. 2017.

[9] H. I. Kobo, A. M. Abu-Mahfouz and G. P. Hancke, "A Survey on Software-Defined Wireless Sensor Networks: Challenges and Design Requirements," IEEE Access, vol. 5, no. , pp. 1872-1899, 2017.

[10] E. Sakic, F. Sardis, J. W. Guck and W. Kellerer, "Towards adaptive state consistency in distributed SDN control plane," 2017 IEEE International Conference on Communications (ICC), Paris, 2017.

[11] D. Wu, D. I. Arkhipov, E. Asmare, Z. Qin and J. A. McCann, "UbiFlow: Mobility management in urban-scale software defined IoT," 2015 IEEE Conference on Computer Communications (INFOCOM), Kowloon, 2015.

[12] J. Liu, Y. Li, M. Chen, W. Dong and D. Jin, "Software-defined internet of things for smart urban sensing," IEEE Communications Magazine, vol. 53, no. 9, pp. 55-63, September 2015.

[13] B. Kang, P. K. H. Nguyen, V. Zalyubovskiy and H. Choo, "A Distributed Delay-Efficient Data Aggregation Scheduling for Duty-Cycled WSNs," IEEE Sensors Journal, vol. 17, no. 11, pp. 3422-3437, June1, 1 2017.

[14] P. Yang, N. Zhang, Y. Bi, L. Yu and X. S. Shen, "Catalyzing Cloud-Fog Interoperation in 5G Wireless Networks: An SDN Approach," in IEEE Network, vol. 31, no. 5, pp. 14-20, 2017.

[15] Deng, Hou, L. Huang, C. Yang, H. Xu, and B. Leng. "Optimizing virtual machine placement in distributed clouds with M/M/1 servers." Computer Communications 102 (2017): 107-119, 2017.

[16] M. G. R. Alam, Y. K. Tun, C. S. Hong, "Multi-agent and Reinforcement Learning Based Code Offloading in Mobile Fog" The International Conference on Information Networking(ICOIN 2016), Jan 13-15, 2016.