TROD: Throughput-Optimal Dynamic Data Traffic Management in Software-Defined Networks

Ayan Mondal, Sudip Misra, and Aishwariya Chakraborty
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur-721302, India
Email: {ayanmondal, sudipm, aishwariya.chakraborty}@iitkgp.ac.in

***Abstract*—In this paper, we study the problem of dynamic data traffic management in software-defined networks (SDNs) in the presence of Internet-of-things (IoT) devices, where IoT devices act as the data traffic sources. In the existing literature, researchers focused on efficient utilization of the ternary content-addressable memory (TCAM) in data traffic management. However, in the presence of IoT-devices, the unbalanced data traffic in SDN deteriorates the overall network performance. Hence, there is a need to design a throughput-optimal dynamic data traffic management scheme for SDN in the presence of IoT-devices, while minimizing the network delay. In this work, we propose an extensive game theory-based scheme, named TROD, for dynamic data traffic management. Additionally, we prove that the dynamic data traffic management, while minimizing the delay and maximizing the throughput of SDN in the presence of IoT-devices, is an NP-complete problem. Hence, we use an evolutionary game theoretic approach to obtain a sub-optimal problem. Thereafter, using linear programming, we obtain the optimal time division matrix, which ensures optimal throughput and delay in SDN. Through simulation, we observe that using TROD, proper distribution of data traffic among the available switches is ensured, and the volumetric overhead per switch reduces by 23.4-29.7%.**

***Index Terms*—Software-Defined Networks, Network Performance, Data traffic, Internet of Things (IoT), Game Theory, Replicator Dynamics.**

I. Introduction

Software-defined networks (SDNs) comprise of the *control* and the *data planes*, among which the network functionalities are divided as — the network control, and the packet forwarding and processing tasks, respectively [1]. There are two types of application programming interfaces (APIs) in control panel such as northbound and southbound APIs. Among these, the southbound APIs are responsible for the controller-switch interaction in SDN architecture. Additionally, the data-traffic in SDN is handled by the southbound APIs with the help of SDN switches. These APIs process the incoming data traffic flow according to the flow-rules installed at the switches. In SDN, the controller installs the flow-rules to the switches for processing the new data traffic flows. On the other hand, the Internet-of-Things (IoT)-devices play a significant role in huge data traffic, i.e., *big-data*, generation in the modern digital world. In the presence of several IoT-devices, SDN is proven to be one of the most promising architecture for managing the generated data traffic [2].

In existing literature, researchers proposed few schemes and architecture for data traffic management in SDN, viz., [1], [3]–[5]. Additionally, the researchers studied the problem of controller and SDN switch placement in existing literature, viz. [2], [4], [6]. However, the problem of unbalanced data traffic in SDN in the presence of IoT devices is overlooked by the researchers. Due to this unbalanced data traffic of IoT devices, the overall performance of the SDN degrades. In other words, due to inefficient data traffic management in SDN, the throughput of the network decreases and the delay in processing the data traffic increases. Hence, in this work, we propose an efficient data traffic management scheme for ensuring optimal delay and throughput in SDN in the presence of IoT devices.

In this paper, we introduce a game theory-based dynamic data traffic management scheme, *named TROD*, for minimizing network delay and maximizing network throughput in SDN in the presence of IoT devices. We use an *evolutionary game-theoretic* approach to deciding the optimal data traffic volume which needs to be handled by the switches, while considering that the data generation rate for each IoT devices is known *a priori*. Moreover, the evolutionary game helps to develop the intermediate sub-optimal problem for efficient data traffic management. Thereafter, with the help of linear programming, we evaluate the *optimal time distribution matrix*, which enables the optimal data traffic distribution in SDN in the presence of IoT-devices. In summary, the specific contributions of this paper are as follows:

1) We present the throughput-optimal data traffic management (TROD) scheme for ensuring minimum delay and maximum throughput of SDN in the presence of IoT-devices.
2) We observe that the dynamic data traffic management is an NP-complete problem. Hence, an evolutionary game theoretic approach is used to obtain the sub-optimal problem which can be solved using linear programming. Subsequently, we evaluate the optimal time distribution matrix for efficient data traffic management.
3) We present an algorithm which includes the controller-switch interaction. The first part of the algorithm deals with the evolutionary game theoretic approach, whereas the last part of the algorithm focuses on evaluating the

optimal time distribution matrix for efficient data traffic management.

II. Related Works

There are several works, which focus on the different aspects of SDN in existing literature. Some of the existing literature, viz. [7]–[9], focus on the theoretical modeling of SDN. On the other hand, some of the existing literature focuses on the data traffic and storage in SDN switches. Some of the existing literature are discussed here. Meiners *et al.* [5] designed a flow-table entry-compressing scheme in order to increase storage space in switch. Congdon *et al.* [3] proposed a latency optimization scheme while reducing the power consumption of the switches. In another work, Mogul *et al.* [10] reduced flow-table lookups in an OpenFlow switch using a hashing based scheme. Reitblatt *et al.* [11] focused on the consistent network update problem, and suggested that each incoming packet follows either the old configuration or the new one while considering a set of abstract operations. Wang *et al.* [12] proposed a SDN-based network storage while having no physical storage. Li *et al.* [6] proposed to store the packet header instead of the entire packet in the buffer of an SDN switch for reducing the communication overhead of SDN. Bera *et al.* [2] provisioned application-aware service in the presence of IoT devices in an SDN-based wireless sensor network. Hayes *et al.* [13] studied the traffic-classification in SDN. Katta *et al.* [4] addressed the trade-off between network update duration and rule-space overhead.

Additionally, some of the works in the existing literature focuses on performance analysis of OpenFlow enabled networks. Jarschel *et al.* [7] studied the OpenFlow architecture as an M/M/1 forward queueing system and an M/M/1-S feedback queueing system. In another work, Metter *et al.* [9] formulated an M/M/∞ queuing model-based analytical model for analyzing a trade-off between signaling rate and switch table occupancy and calculate an optimum flow-rule time-out period.

***Synthesis**:* We infer that there exist few research work on data traffic management in SDN while utilizing storage of the switch, i.e., the ternary content-addressable memory (TCAM), efficiently. However, no scheme is proposed to deal with the problem of unbalanced data traffic in SDN in the presence of IoT devices. Additionally, throughput-optimal dynamic data traffic management in SDN while ensuring minimal network delay is in demand.

III. System Model

We consider a software-defined network (SDN) having a single controller and multiple SDN switches. The schematic diagram of SDN architecture is shown in Figure 1. We consider that the IoT devices are connected with the SDN switches through the access point (APs). These IoT devices are heterogeneous in nature with different data generation rate. We consider a stochastic process, where the IoT devices are static for a fixed time duration. Each AP is connected with *multiple* SDN switches. The data traffic generated by the IoT devices are forwarded to one of the connected SDN switches via available AP. Thereafter, the data traffic is processed by the SDN switches according to the flow-rule entries and forwarded to the backhaul network for further processing.

In case of table-miss, meta-data of the data traffic is forwarded to the controller, and it installs the corresponding rule to one of the switches. Additionally, due to limited TCAM, there is a limitation of the maximum number of rules that can be installed in an SDN switch. For example, according to the OpenFlow specification version 1.5.0 [14], maximum 8000 rules can be installed in an OpenFlow switch, while considering that 45 match fields are present in each rule in the flow table.

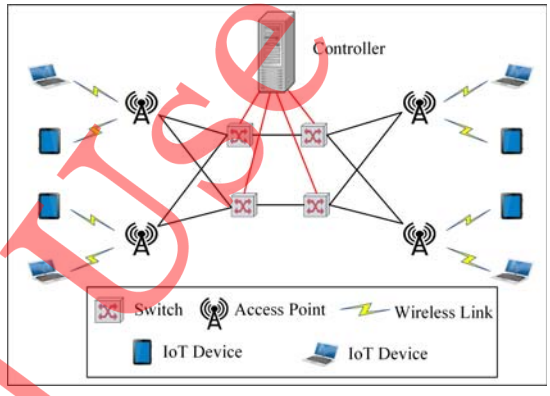


Fig. 1: Schematic Diagram of SDN in the Presence of IoT-Devices

As shown in Figure 1, with the change in the position of edge devices, i.e., IoT devices, the installed rules need to be changed dynamically by the controller. In such a situation, the throughput of the SDN switch gets affected depending on the number of active flow-rules in the flow table and the volume of the data traffic. We define *active flow-rule* as mentioned in Definition 1.

Definition 1. *We define a flow-rule to be active at time instant t if that rule is used for forwarding data traffic within a time duration of $[t-\delta, t]$, where $\delta > 0$.*

We consider that an IoT-enabled SDN comprises of a single controller, multiple SDN switches, and multiple IoT devices. Each switch $s \in \mathcal{S}$, where $\mathcal{S}$ is the set of available switches, is capable of installing maximum R_s^{max} number of rules. Therefore, the maximum number of rules $\mathbb{R}^{max}$, which can be accommodated is expressed as follows:

$$\mathbb{R}^{max} = \sum_{s \in \mathcal{S}} R_s^{max} \tag{1}$$

Moreover, we consider that each IoT device $n \in \mathcal{N}$, where $\mathcal{N}$ is the set of available IoT devices in SDN, generates f_n number of data traffic flows. The data generation-rate of each data traffic flow i, where $0 \leq i \leq f_n$ and $n \in \mathcal{N}$, is denoted as λ_i. Additionally, we consider that the volume of data associated with each data traffic flow i, where $0 \leq i \leq f_n$ and $n \in \mathcal{N}$, is denoted by ν_i. Therefore, we have:

$$\nu_i = \sum_{t=1}^{T} \lambda_i \tag{2}$$

We consider that each data traffic flow i generated by the IoT devices is *mutually exclusive*. Therefore, the data traffic management by the controller needs to ensure the following constraint:

$$\sum_{n \in \mathcal{N}} f_n \leq \mathbb{R}^{max} \tag{3}$$

We consider that each switch $s \in \mathcal{S}$ handles F_s number of flow-rules. Additionally, the incoming-flow rate at time instant t for each switch s is denoted as $\lambda_s(t)$. Hence, the incoming volume of data traffic V_s handled by each switch s is defined as follows:

$$V_s = \sum_{j=1}^{F_s} \nu_j = \sum_{t=1}^{T} \lambda_s(t) \tag{4}$$

where T is the time duration for which the IoT devices remain static. On the other hand, each switch s has a processing limit of μ_s amount of data per unit time. Therefore, in order to ensure optimal throughput of the network, the controller needs to ensure that the data traffic is properly distributed among the available switches.

IV. Throughput-Optimal Data Traffic Management (TROD) Scheme

A. Justification for Using Evolutionary Game

For efficient data traffic management, we need to ensure that the load of the network, i.e., the volumetric data, is optimally distributed among the available switches. Hence, in order to reduce the processing delay in the switch, the controller tries to reduce the load on each switch. However, the controller also needs to ensure the overall throughput of the network. Thereby, our objective is as follows:

$$\max_{\sum \mu_s} \min_{\lambda_s} \sum_{s \in \mathcal{S}} V_s \tag{5}$$

while satisfying the constraints given in Equations (3) and (4). Additionally, we need to satisfy the following constraints:

$$\lambda_s \geq 0 \text{ and } \mu_s \leq \mu^{max} \tag{6}$$

$$\sum_{s \in \mathcal{S}} x_{s,i}(t) = 1 \quad \text{and} \quad F_s = \sum_{n \in \mathcal{N}} \sum_{i=1}^{f_n} x_{s,i} \tag{7}$$

where μ^{max} is the maximum processing rate of the SDN switches; and $x_{s,i}(t)$ denotes the association vector of flow-rule F_s, and is a binary variable. We define:

$$x_{s,i}(t) = \begin{cases} 1, & \text{if Rule } i \text{ is installed at switch } s \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Therefore, we argue that the aforementioned problem is an *integer programming* problem, in which the variable $x_{s,i}(t)$ can take only binary values — $\{0, 1\}$. Hence, this problem is an *NP-complete* problem [15]. Thereby, we use an evolutionary game theoretic approach in order to reduce it to a *linear programming* problem, and achieve a sub-optimal solution, i.e., ensuring optimal throughput of the network, in polynomial time.

B. Game Formulation

In order to achieve a sub-optimal solution, we use an *evolutionary game theoretic* approach. In the proposed dynamic data traffic management scheme, named *TROD*, the IoT devices act as the players and choose the set of optimal SDN switches, i.e., strategies in TROD, for forwarding data with the help of SDN controller. Here, the controller acts as a centralized coordinator. In TROD, we consider that the population share, i.e., the total volume of data traffic, needs to be divided among the available switches. Therefore, the population share $y_s(\omega)$ of each switch $s \in \mathcal{S}$ is defined as follows:

$$y_s(\omega) = \frac{V_s(\omega)}{\sum\limits_{s \in \mathcal{S}} V_s} \tag{9}$$

where ω is the evolutionary iteration.

Utility Function of Each Switch: Utility value $\phi_s(\omega)$ signifies the fitness function for switch s. We consider that $\phi_s(\omega)$ varies linearly with the population share of switch s. On the other hand, with the increase in number of active flow-rules $F_s(\omega)$ installed, the utility value decreases. For example, switches s_1 and s_2 have population share of $y_{s_1}(\omega)$ and $y_{s_2}(\omega)$, respectively. Additionally, the number of rules installed in the switches s_1 and s_2 are $F_{s_1}(\omega)$ and $F_{s_2}(\omega)$, respectively. Hence, considering that $F_{s_1}(\omega) = F_{s_2}(\omega)$, we get $\phi_{s_1}(\omega) \succeq \phi_{s_2}(\omega)$, where $y_{s_1}(\omega) \geq y_{s_2}(\omega)$. On the other hand, we observe $\phi_{s_1}(\omega) \preceq \phi_{s_2}(\omega)$, while considering that $F_{s_1}(\omega) \geq F_{s_2}(\omega)$ and $y_{s_1}(\omega) = y_{s_2}(\omega)$. Hence, we define the utility function as follows:

$$\phi_s(\omega) = y_s(\omega) \left(1 - \frac{F_s(\omega)}{R_s^{max}}\right) \tag{10}$$

Thereafter, the controller calculates average payoff $\overline{\phi}(\omega)$ of the population using the following equation:

$$\overline{\phi}(\omega) = \sum_{s \in \mathcal{S}} y_s(\omega) \phi_s(\omega) \tag{11}$$

Replicator Dynamics of TROD: In TROD, each switch acts as a replicator, and tries to mutate and evolve over successive iteration and reach the evolutionary equilibrium. Dynamic of the change in population share is obtained using *replicator dynamics*. In TROD, the replicator dynamics of each switch s is represented as follows:

$$\dot{y_s}(\omega) = \sigma y_s(\omega) \left(\phi_s(\omega) - \overline{\phi}(\omega)\right) \tag{12}$$

where σ is the evolution controlling factor. In TROD, σ value determines the permissible rate of change in population share in two successive evolutionary iterations.

Reduced Sub-Optimal Problem: By using evolutionary game theoretic approach, we get the population share y_s^* of each switch s at equilibrium, while satisfying the following constraint:

$$\dot{y_s}(\omega)|_{y_s(\omega)=y_s^*} \approx 0 \tag{13}$$

Hence, the vector of the volume of data traffic, which needs to be handled by the switches, is defined as follows:

$$\boldsymbol{V} = \boldsymbol{y} \sum_{s \in \mathcal{S}} V_s \tag{14}$$

where $\boldsymbol{V} = [V_1^* \ \cdots \ V_{|\mathcal{S}|}^*]^T$ and $\boldsymbol{y} = [y_1^* \ \cdots \ y_{|\mathcal{S}|}^*]^T$.

Therefore, we yield a system of linear equations, which is easily solvable using linear programming. The system of linear equations is expressed as follows:

$$\boldsymbol{\lambda} \mathbb{T}^T = \boldsymbol{V} \tag{15}$$

where $\boldsymbol{\lambda} = [\lambda_1 \ \cdots \ \lambda_{(\sum\limits_{n \in \mathcal{N}} f_n)}]$, and $\mathbb{T}^T$ defines the transpose of matrix $\mathbb{T}$. We define time-distribution matrix $\mathbb{T}$ as follows:

$$\mathbb{T} = \begin{bmatrix} \Delta t_{11} & \cdots & \Delta t_{(\sum f_n)1} \\ \vdots & \ddots & \vdots \\ \Delta t_{1|\mathcal{S}|} & \cdots & \Delta t_{(\sum f_n)|\mathcal{S}|} \end{bmatrix} \tag{16}$$

where Δt_{is} defines the time duration for which data traffic flow i gets forwarded by switch s. Additionally, Equation (15) needs to satisfy the following constraint:

$$\sum_{n \in \mathcal{N}} \sum_{i=1}^{f_n} \Delta t_{is} = T, \quad \forall s \in \mathcal{S} \tag{17}$$

C. Theoretical Analysis

In this section, we analyze theoretically the proposed scheme, TROD, while evaluating the existence of evolutionary equilibrium. As mentioned earlier, at evolutionary equilibrium, the change in population share $\dot{y_s}(\omega)$ of each switch s becomes zero. Therefore, we get:

$$\sigma y_s(\omega) \left(\phi_s(\omega) - \overline{\phi}(\omega)\right) = 0 \tag{18}$$

We consider that $\sigma > 0$ and is a constant, and each switch s has a positive population share, i.e., $y_s(\omega) > 0$. Therefore, Equation (18) can be written as follows:

$$\phi_s(\omega) - \overline{\phi}(\omega) = 0 \tag{19}$$

By solving Equation(19), we get:

$$(y_s^*)^2 - y_s^* + \frac{\sum\limits_{s' \in \mathcal{S}/\{s\}} (y_{s'}^*)^2 \left[1 - \frac{F_{s'}}{R_{s'}^{max}}\right]}{1 - \frac{F_s}{R_s^{max}}} = 0 \tag{20}$$

Hence, theoretically, we yield that the optimal population share y_s^* of each switch s is as follows:

$$y_s^* = \frac{1 \pm \sqrt{1 - 4\zeta}}{2} \tag{21}$$

where $\zeta = \left[\frac{\sum\limits_{s' \in \mathcal{S}/\{s\}} (y_{s'}^*)^2 \left[1 - \frac{F_{s'}}{R_{s'}^{max}}\right]}{1 - \frac{F_s}{R_s^{max}}}\right]$. Therefore, we argue that the proposed scheme, TROD, ensures evolutionary equilibrium. Based on the derived $\{y_s^* | \forall s \in \mathcal{S}\}$ values, we get $(|\mathcal{S}| + \sum\limits_{n \in \mathcal{N}} f_n)$ number of linear equations. By solving the linear equations, we can easily obtain the dynamic data traffic distribution over a fixed time duration T, while ensuring optimal delay at the switch-end and optimal throughput of SDN.

Algorithm 1 Algorithm for TROD Scheme

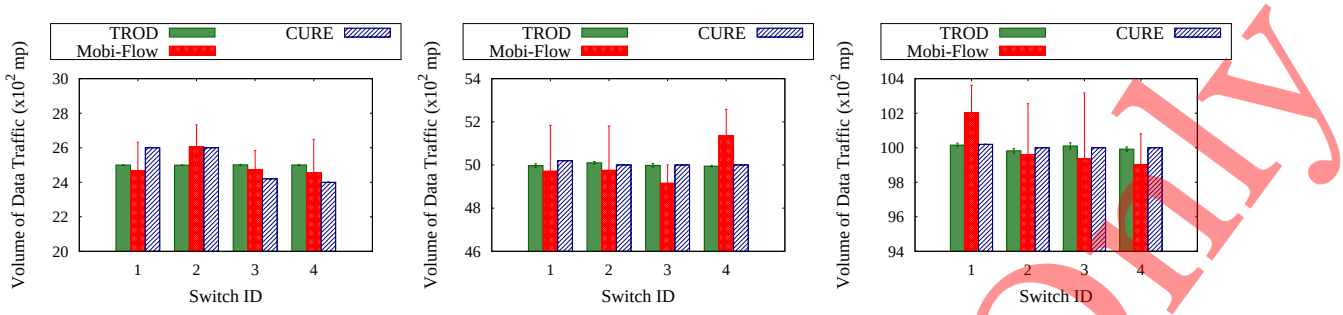
INPUTS:
- $\mathbb{N}$ ▷ Set of available IoT devices
- $\mathcal{S}$ ▷ Set of available SDN switches
- f_n ▷ Number of flows of IoT-device n
- F_s ▷ Flow-rules installed in switch s
- $\mathbb{R}^{max}$ ▷ Maximum number of rules in SDN
- R_s^{max} ▷ Maximum number of rules in switch s
- λ_i ▷ Generation-rate of data traffic flow i
- σ ▷ Evolution controlling factor

OUTPUT:
- $\mathbb{T}$ ▷ Time-distribution matrix

PROCEDURE:
1: $\omega \leftarrow 1$
2: Randomly map each data traffic flow $0 \leq i \leq f_n$, where $n \in \mathcal{N}$ to any switch $s \in \mathcal{S}$
3: **for** Each $s \in \mathcal{S}$ **do**
4: Calculate $V_s(\omega)$ using Equations (2) and (4)
5: Calculate population share $y_s(\omega)$ using Equation (9)
6: **end for**
7: **do**
8: **for** Each $s \in \mathcal{S}$ **do**
9: Calculate utility value $\phi_s(\omega)$ using Equation (10)
10: **end for**
11: Calculate average payoff $\overline{\phi}(\omega)$ of the population using Equation (11)
12: **for** Each $s \in \mathcal{S}$ **do**
13: Calculate replicator dynamics $\dot{y_s}(\omega)$ using Equation (12)
14: $\omega \leftarrow \omega + 1$
15: $y_s(\omega) \leftarrow y_s(\omega - 1) - \dot{y_s}(\omega - 1)$
16: **end for**
17: **while** $(\dot{y_s}(\omega) \approx 0)$
18: $y_s^* \leftarrow y_s(\omega)$
19: Calculate $\boldsymbol{V}$ using Equation (14)
20: Calculate $\mathbb{T}$ using Equation (15)
21: **return** $\mathbb{T}$;

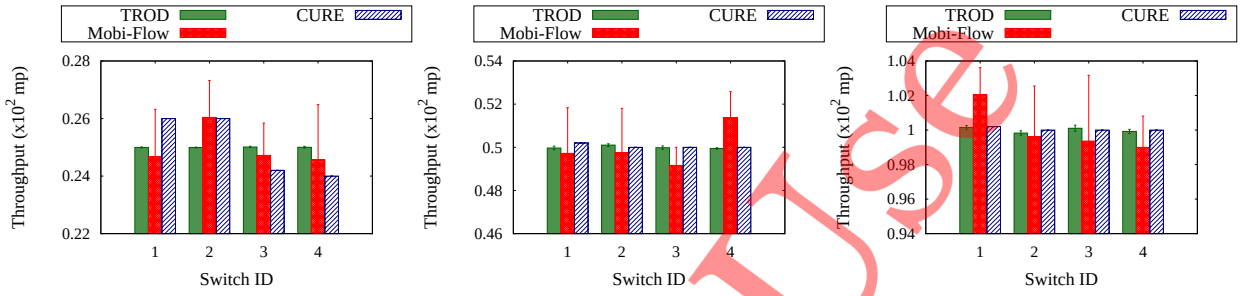
D. Proposed Algorithm

In order to achieve the sub-optimal solution of TROD, the controller tries to distribute the data-traffic among the available SDN switches. Thereby, the controller tries to ensure that each switch satisfies the equilibrium condition. The dynamic data traffic management scheme, TROD, ensures delay and throughput-optimal data traffic in IoT-enabled SDN using Algorithm 1. In Algorithm 1, each data traffic flow i, where $1 \leq i \leq f_n$ and $n \in \mathcal{N}$, is mapped randomly to one of



(a) Number of IoT-Device = 50 (b) Number of IoT-Device = 100 (c) Number of IoT-Device = 200

Fig. 2: Volume of Data Traffic Processed by Switches with Varied Number of IoT Devices



(a) Number of IoT-Device = 50 (b) Number of IoT-Device = 100 (c) Number of IoT-Device = 200

Fig. 3: Throughput of Switches with Varied Number of IoT Devices

the available switches. Thereafter, with the help of controller, each switch decides the optimal population share, i.e., data-traffic volume needed to be handled over the time-period T. Finally, using the linear equations derived with the help of evolutionary game theory, the controller schedules the data-traffic among the available switches with the help of time-distribution matrix.

V. Performance Evaluation

In this section, we analyze the performance of the proposed scheme, TROD, with the varying number of data traffic flows and the available switches in SDN. Generic test-bed information for TROD is provided in Table I.

TABLE I: System Specification

Parameter	Value
Processor	Intel(R) Core(TM) i5-2500 CPU @ 3.30 GHz
RAM	4 GB DDR3
Disk Space	500 GB
Operating System	Ubuntu 16.04 LTS
Application Software	MATLAB 2015b

A. Simulation Parameters

For simulation, we varied the number of SDN switches and the number of IoT-devices as mentioned in Table II. We simulated the proposed scheme in MATLAB simulation platform, as mentioned in Table I. We consider that each flow generates data traffic at the rate of 0.2 million packets per second ($mpps$). We simulate TROD for 100 simulation seconds.

B. Benchmark

The performance of the proposed scheme, TROD, is evaluated by comparing with existing schemes — Mobi-Flow [2] and CURE [16]. In Mobi-Flow, Bera *et al.* [2] proposed a data traffic management scheme in the presence of IoT-devices while predicting the location of the IoT-devices beforehand. The authors also studied the dynamic data traffic management, while installing the flow-rules in the SDN switches, dynamically. On the other hand, In CURE, Maity *et al.* [16] proposed a data traffic management scheme while considering the update of flow-rules in the switches according to the priorities of the switches.

TABLE II: Simulation Parameters

Parameter	Value
Number of SDN switches	$4, 8, 12$
Number of IoT-devices	$50, 100, 150, 200$
Number of flow per device	10
Data traffic generation rate	$0.2\ mpps/flow$
Maximum number of flow rules per switch	8000 [14]
Simulation duration	$100\ sec$

C. Performance Metrics

We evaluated the performance of the proposed scheme, TROD, using the following metrics:

Volume of data traffic processed by each switch: We consider that with the increase in the volume of data processed by each switch, the queuing delay increases. Hence, we consider the volume of data traffic processed by each switch.

Throughput of each switch: The overall throughput of the network depends linearly on the throughput of the individual switch. On the other hand, throughput per switch also signifies the balance factor of the switches in data traffic management.

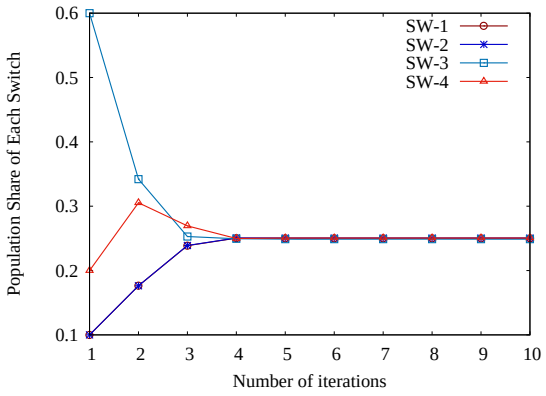


Fig. 4: Population Share of each Switch

D. Results and Discussions

For simulations, we consider that the controller updates the flow-tables of each switch, periodically in every 10 simulation seconds. From Figure 2, we observe that with the increase in the number of IoT-devices, the volume of data traffic increases. However, using TROD, the volume of data traffic gets distributed equally among the available switches, than using Mobi-Flow and CURE. We argue that the proposed scheme, TROD, distributes the incoming traffic among the available switches equally. Hence, the queuing delay of the network reduces by 23.4-29.7%.

On the other hand, from Figure 3, we observe that the throughput of each switch is the same using TROD. Hence, we argue that the data traffic in the network is distributed properly using TROD, than using Mobi-Flow and CURE. From Figure 3, we observe that the throughput of each switch increases by 23.1-28.78% using TROD than using Mobi-Flow and CURE. Additionally, from Figure 4, we observe that the change in population share of each switch varies linearly with the difference between the actual payoff value of the switch and the average payoff value of the population. Moreover, we observe that within 4-6 iterations, TROD reaches its evolutionary equilibrium.

Therefore, we argue that the proposed scheme, TROD, enhances the performance of network holistically, i.e., minimizes the network delay and maximizes the network throughput, while distributing the incoming data traffic, efficiently and dynamically, in SDN in the presence of IoT-devices.

VI. Conclusion

In this paper, we observe that dynamic data traffic management in SDN is an NP-complete problem. Therefore, we formulated an evolutionary game theory-based TROD scheme to obtain sub-optimal problem for data traffic management in SDN in the presence of IoT-devices. We observe that the proposed scheme, TROD, ensures proper distribution of data traffic among the switches while minimizing the network delay and maximizing the throughput of the network. Moreover, through simulations, we observe that TROD outperforms the existing schemes – Mobi-Flow and CURE.

Future extension of this work includes an understanding of data traffic distribution with multihop data flow in SDN in the presence of IoT-devices. Additionally, this work also can be extended while incorporating the mobility of IoT-devices in SDN.

Acknowledgment

Ayan Mondal acknowledges TCS Fellowship.

References

[1] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," in *Proc. of the IEEE*, vol. 103, no. 1, January 2015, pp. 14–76.

[2] S. Bera, S. Misra, S. K. Roy, and M. S. Obaidat, "Soft-WSN: Software-Defined WSN Management System for IoT Applications," *IEEE Syst. J.*, pp. 1–8, 2016, DOI:10.1109/JSYST.2016.2615761.

[3] P. T. Congdon, P. Mohapatra, M. Farrens, and V. Akella, "Simultaneously Reducing Latency and Power Consumption in OpenFlow Switches," *IEEE/ACM Trans. on Net.*, vol. 22, no. 3, pp. 1007–1020, June 2014.

[4] N. P. Katta, J. Rexford, and D. Walker, "Incremental Consistent Updates," in *Proc. of ACM SIGCOMM Wrkshp*. New York, NY, USA: ACM, 2013, pp. 49–54.

[5] C. R. Meiners, A. X. Liu, and E. Torng, "Bit Weaving: A Non-Prefix Approach to Compressing Packet Classifiers in TCAMs," *IEEE/ACM Trans. on Net.*, vol. 20, no. 2, pp. 488–500, April 2012.

[6] F. Li, J. Cao, X. Wang, Y. Sun, T. Pan, and X. Liu, "Adopting SDN Switch Buffer: Benefits Analysis and Mechanism Design," in *Proc. of IEEE ICDCS*, Jun 2017, pp. 2171–2176.

[7] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia, "Modeling and Performance Evaluation of an OpenFlow Architecture," in *Proc. of Int. Tel. Cong.* International Teletraffic Congress, 2011, pp. 1–7.

[8] A. Mondal, S. Misra, and I. Maity, "Buffer Size Evaluation of OpenFlow Systems in Software-Defined Networks," *IEEE Syst. J.*, pp. 1–8, 2018.

[9] C. Metter, M. Seufert, F. Wamser, T. Zinner, and P. Tran-Gia, "Analytical Model for SDN Signaling Traffic and Flow Table Occupancy and Its Application for Various Types of Traffic," *IEEE Trans. on Net. and Ser. Man.*, vol. 14, no. 3, pp. 603–615, Sep 2017.

[10] J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, A. R. Curtis, and S. Banerjee, "DevoFlow: Cost-effective Flow Management for High Performance Enterprise Networks," in *Proc. of ACM SIGCOMM Wrksp*, New York, NY, USA, 2010, pp. 1–6.

[11] M. Reitblatt, N. Foster, J. Rexford, C. Schlesinger, and D. Walker, "Abstractions for Network Update," in *Proc. of ACM SIGCOMM*, New York, NY, USA, 2012, pp. 323–334.

[12] M.-H. Wang, P.-W. Chi, J.-W. Guo, and C.-L. Lei, "SDN storage: A Stream-Based Storage System over Software-Defined Networks," in *Proc. of IEEE INFOCOM Wrksps*, Apr 2016, pp. 598–599.

[13] M. Hayes, B. Ng, A. Pekar, and W. K. G. Seah, "Scalable Architecture for SDN Traffic Classification," *IEEE Syst. J.*, pp. 1–12, 2017, DOI: 10.1109/JSYST.2017.2690259.

[14] OpenFlow. (2014, December) OpenFlow Switch Specification Version 1.5.0. Open Networking Foundation. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf

[15] J. D. Ullman, "NP-complete scheduling problems," *J. of Com. & Syst. Sc.*, vol. 10, no. 3, pp. 384–393, 1975.

[16] I. Maity, A. Mondal, S. Misra, and C. Mandal, "CURE: Consistent Update with Redundancy Reduction in SDN," *IEEE Trans. on Comm.*, vol. PP, no. 99, pp. 1–8, 2018.