A Dormant Multi-Controller Model for Software Defined Networking

FU Yonghong[1,2,3], BI Jun[1,3],WU Jianping[1,2,3], CHEN Ze[1,2,3], WANG Ke[4], LUO Min[5]

[1] Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China
[2] Department of Computer Science, Tsinghua University, Beijing 100084, China
[3] Tsinghua National Laboratory for Information Science and Technology (TNList), Beijing 100084, China
[4] Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
[5] Shannon Lab, Huawei Technologies Company, Santa Clara 950950, USA

Abstract: In order to improve the scalability and reliability of Software Defined Networking (SDN), many studies use multiple controllers to constitute logically centralized control plane to provide load balancing and fail over. In this paper, we develop a flexible dormant multi-controller model based on the centralized multi-controller architecture. The dormant multi-controller model allows part of controllers to enter the dormant state under light traffic condition for saving system cost. Meanwhile, through queueing analysis, various performance measures of the system can be obtained. Moreover, we analyze the real traffic of China Education Network and use the results as the parameters of computer simulation and verify the effects of parameters on the system characteristics. Finally, a total expected cost function is established, and genetic algorithm is employed to find the optimal values of various parameters to minimize system cost for the deployment decision making.

Keywords: Software Defined Networking; multi-controller; queueing theory; performance evaluation

I. INTRODUCTION

Software Defined Networking (SDN) is an emerging network architecture where network control is decoupled from forwarding and is directly programmable [1]. Through SDN, network protocol can be innovated, and new service can be deployed based on different devices.

The centralized control plane with single controller is the bottleneck of SDN when thousands of high-speed OpenFlow switches work in the data plane. In a OpenFlow switch, the forwarding policy is stored in flow table. When new flow arrives, the OpenFlow switch matches the header of the packet with flow entries and sends the unmatched packet to the controller with a Packet-In message [2]. The controller is responsible for dealing with the received Packet-In messages by setting up flow mods to all switches in the forwarding path. Unfortunately, the processing ability of a single controller cannot meet the processing requirements of large-scale network. We analyze the real traffic of China Education Network, and find that the maximal number of flows per second is about 3 million. In such a situation, the control plane will be under tremendous pressure due to the capacity limitation of flow entries in TCAM. The throughput of NOX with a single thread is about 21000 messages per second [3]. By applying the multi-threading technique, Maestro improves

the throughput of SDN controller, but the throughput of the controller is still far away from 3 million [3]. Meanwhile, Tavakoli et al. [4] indicate that the CPU of single NOX controller is the bottleneck under large network topology. In their 2 million VMs topology, the control plane would require 667 controllers.

In order to improve the scalability and reliability of SDN, many researchers use multiple controllers to constitute a logically centralized control plane, such as Ethane, HyperFlow, Onix, DIFANE and Kandoo. Ethane adopt multiple controllers to provide fault tolerance [5]. HyperFlow present a distributed event-based control plane for OpenFlow. In HyperFlow, all the controllers share the same consistent network-wide view through a publish/subscribe system [6]. Onix develops a distributed system, which runs on a cluster of one or more physical servers, each of which may run multiple Onix instances works like multiple controllers [7]. DIFANE employs authority switches to store necessary rules as multiple controllers [8]. Kandoo has a two layer multi-controller architecture. The bottom layer of it is a group of distributed independent controllers, and the top layer of it is a centralized root controller that maintains a global network state [9]. The above researches are mainly focused on architecture design and implementation. ElastiCon considers the traffic conditions can depend on the time of day, and the load may not balance in different controllers. They realized that one key limitation of the above systems is that the mapping between the data plane switches and the controllers are statically configured. So they propose an elastic architecture in which the controller pool is dynamically grown or shrunk according to given threshold values, and introduce a novel switch migration protocol for enabling seamless load shifting [10]. SOX proposes a centralized multi-controller architecture, which enables all controllers provide autonomous load balancing and fail over and supports dynamically adjust the number of controllers in the control plane [11]. However, the existing researches such as ElastiCon and SOX, which can dynamically adjust the number of controllers, are based on given threshold values. They did not discuss whether the threshold values are optimal values. Meanwhile, they use qualitative analysis method rather than quantitative analysis method to evaluate the performance measures. This means that when the network size changes, we cannot determine how many controllers should be employed and what the optimal threshold value is. Moreover, the performance under different number of working controllers is also unknown.

In this paper, we propose a dormant multi-controller model for SDN. The highlights of this paper include: (1) The proposed dormant multi-controller model allows part of idle controllers to enter the dormant state under light traffic condition for saving system cost. (2) We use quantitative analysis method to evaluate the performance of the system. Detailed speaking, *M/M/c* queue with *N* policy and (*d*, *c*) vacation mechanism is used to analyze this model. Through queueing analysis, various performance measures of the system are obtained. (3) Utilizing the key performance measures, we establish an expected total cost function. Based on the cost function, genetic algorithm is employed to find the optimal designing variables, so as to minimize the system cost. (4) We analyze the real traffic of China Education network, and use the results as the input of computer simulation and genetic algorithm. The optimal designing variables obtained from the genetic algorithm can be used in the deployment decision making. To our knowledge, there has been no such research in SDN till now.

The dormant multi-controller model is based on a general centralized multi-controller architecture (shown in Fig.1). In the centralized multi-controller architecture, global network state is maintained in a central Network Information Base (NIB), and all controllers in the controller cluster adopt Equal Mode, then if one controller collapse the remaining controllers will responsible for the total work

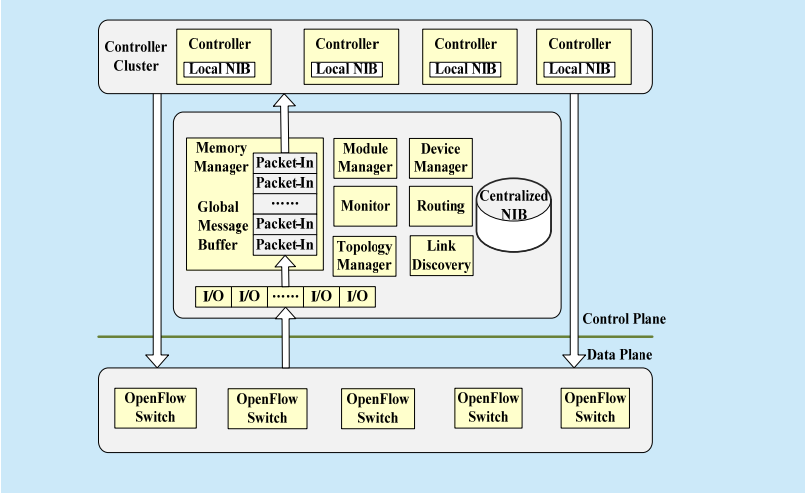

Fig.1 *Centralized multi-controller architecture*

load. The centralized multi-controller architecture can guarantee global consistency, costs shorter convergence time when changes happen in the network, and it is easy to management.

The rest of this paper is organized as follows. In Section II, we give model description and some notations. In Section III, using the matrix analytic method, we derive the steady state probability vector. Section IV is devoted to the performance measures of the system. In Section V, we write a simulation program to validate our theoretical analysis. In Section VI, genetic algorithm is employed to find the optimal values of various parameters to minimize system cost for decision making. Finally, conclusions are given in Section VII.

II. Model Description

We assume that the packets arrive at the edge switches are according to Poisson process with rate $\lambda_i(\lambda_i > 0)$, and the sum of independent Poisson processes is also a Poisson process with rate equals to the sum of rates of the independent Poisson processes. We assume the arrival rate of the aggregated stream is according to Poisson process with parameter $\lambda\ (\lambda = \sum_{i=1}^{n} \lambda_i * p_i)$. If the service rate is less than the arrival rate $(\mu < \lambda)$, then the departure rate is less than the arrival rate that will result in control plane congestion and packets loss. In this situation, we adopt multiple controllers (c controllers, let $c\mu > \lambda$) to solve this situation. The centralized SDN multi-controller model is shown in Fig.2.

We consider using a continuous-time *M/M/c* queueing system with *N* policy and (*d*, *c*) mechanism to analyze the dormant multi-controller model of SDN. In queueing system, *N*-policy is a threshold policy investigated by Yadin and Naor [12] used to control the service. Heyman [13] proves that *N* policy is the optimal strategy for operating *M/G/1* queues under various cost criteria. The *N* policy in our model works as follows: when the queue length in the system increases and hits a threshold value *N*, the system active all dormant controllers to provide service. Shen and Jin [14] introduced *M/M/c* queue with *N* policy and multiple vacation of partial servers based on matrix geometric solutions. Ke et al. [15] considered a finite buffer *M/M/c* queueing system in which servers are unreliable and follow a (*d*, *c*) vacation mechanism. The (*d*, *c*) vacation mechanism in the model works as follows: The number of controllers in the system is $c\ (c\mu > \lambda)$, once the queue length decrease to 0, and all packets assigned to *d* controllers complete their service, the *d* idle controllers will go to a dormant state to reduce system cost.

The following is the model description. Assume Packet-In messages arrive at the control plane according to a Poisson process with parameter $\lambda\ (\lambda > 0)$ and the Packet-In messages service time of each controller follows an exponential distribution with parameter $\mu\ (\mu > 0)$. The system capacity is finite, denoted by $K(K < \infty)$. Once the queue length decreases to 0, and all messages assigned to the *d* controllers complete their service, the *d* idle controllers will go to the dormant state to reduce system cost and improve the utilization of other working controllers. In the event that the queue length in the system hits a pre-determined threshold *N*, all dormant controllers will return to normal working state, and the

dormant return time of controllers is exponential distributed with parameter . If the number of messages in the system reaches the system capacity limit K, then no more messages are allowed to enter the system until the messages being served and the number of messages in the system decreases less than K. If we want to avoid the appearing of bursty traffic, we can set a big buffer capacity K to cache the peak traffic to decrease the loss probability, and the message delay time will increase along with the increasing message length of buffer. But if the departure rate is higher than the arrival rate ($c\mu > \lambda$), the message delay time will not increase even we set a big K.

III. Queueing Analysis

The following random variables describe the system state, namely

N(t) =the number of messages in the system at time t,

$$\mathrm{Y(t)}=\begin{cases}0, & c-d \text{ controllers on the working state}\\ & \text{and d controllers on the dormant state,}\\ 1, & c \text{ controllers on the working state.}\end{cases}$$

It is obvious that the 2-dimension process {N(t), Y(t)} is the irreducible regular continuous time Markov chain. Consider

$$\Delta_0=\{(i,\ 0):N(t)=i,\ Y(t)=0|i=0,1,\cdots,K\},$$

$$\Delta_1=\left\{(i,\ 1):N(t)=i,\ Y(t)=1|\begin{matrix}i=c-d+1,\\ c-d+2,\cdots,K\end{matrix}\right\}.$$

The state space for this queueing system can be described as Δ, where $\Delta=\Delta_0\cup\Delta_1$.

By partitioning the state space into levels with respect to the number of messages in the system, we find that the corresponding generator matrix **Q** of the above Markov process exhibiting the following block-structured form:

$$\mathbf{Q}=\begin{pmatrix}A_{0,0} & A_{0,1} & & & & \\ A_{1,0} & A_{1,1} & \ddots & & & \\ & \ddots & \ddots & \ddots & & \\ & & A_{K-1,K-2} & A_{K-1,K-1} & A_{K-1,K} \\ & & & A_{K,K-1} & A_{K,K}\end{pmatrix}$$

Where

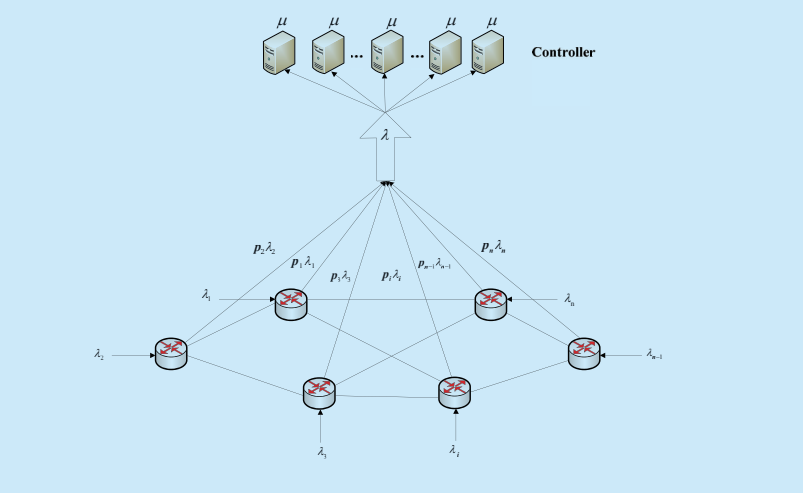

Fig.2 *SDN multi-controller model*

$$A_{i,i}=\begin{cases}-(\lambda+i\mu),\ 0\leqslant i\leqslant c-d,\\ \begin{pmatrix}-(\lambda+(c-d)\mu) & 0\\ 0 & -(\lambda+i\mu)\end{pmatrix},\\ c-d+1\leqslant i\leqslant c-1,\\ \begin{pmatrix}-(\lambda+(c-d)\mu) & 0\\ 0 & -(\lambda+c\mu)\end{pmatrix},\\ c\leqslant i\leqslant N-1,\\ \begin{pmatrix}-(\lambda+(c-d)\mu+\theta) & \theta\\ 0 & -(\lambda+c\mu)\end{pmatrix},\\ N\leqslant i\leqslant K-1,\\ \begin{pmatrix}-((c-d)\mu+\theta) & \theta\\ 0 & -c\mu\end{pmatrix},i=K,\end{cases}$$

$$A_{i,i-1}=\begin{cases}(i\mu),\ 0\leqslant i\leqslant c-d,\\ \begin{pmatrix}(c-d)\mu\\ i\mu\end{pmatrix},i=c-d+1,\\ \begin{pmatrix}(c-d)\mu & 0\\ 0 & i\mu\end{pmatrix},\\ c-d+2\leqslant i\leqslant c-1,\\ \begin{pmatrix}(c-d)\mu+\theta & 0\\ 0 & c\mu\end{pmatrix},c\leqslant i\leqslant K,\end{cases}$$

$$A_{i,i+1}=\begin{cases}(\lambda),0\leqslant i\leqslant c-d-1,\\ \begin{pmatrix}\lambda & 0\end{pmatrix},i=c-d,\\ \begin{pmatrix}\lambda & 0\\ 0 & \lambda\end{pmatrix},\\ c-d+1\leqslant i\leqslant K-1.\end{cases}$$

For convenience, let us combine these probabilities into the row-vectors.

$\pi_i=(\pi_{i,0}),\ i=0,\ 1,\ \cdots,\ c-d,$

$\pi_i=(\pi_{i,0},\ \pi_{i,1}),\ i=c-d+1,\ \cdots,\ K.$

It is well known that the vector $\pi=(\pi_0,\pi_1,\cdots,\pi_K)$ is the unique solution to the following system of linear algebraic equations.

$$\begin{cases} \pi\mathbf{Q} = \mathbf{0}, \\ \pi\mathbf{e} = 1. \end{cases} \quad (1)$$

where **e** denotes the column vector of appropriate size consisting of ones. Here and in the sequel, **0** is the zero row vector with appropriate dimension.

If K is very large, the direct solving of the system (1) can be time and resource consuming. We use an effective and numerically stable algorithm based on matrix analytic method for solving this system provided in Ref. [16]. This algorithm is described in the following Theory.

Theorem 1. The stationary probability vectors i, i =1, 2, . . . ,K, are calculated as follows:

$$\begin{cases} \pi_i = (-1)^i \pi_0 \prod_{s=1}^{i} A_{s-1,s}(A_{s,s} + A_{s,s+1}G_s)^{-1}, \\ i = 1, 2, ..., K-1, \\ \pi_K = (-1)^K \pi_0 \prod_{s=1}^{K-1} A_{s-1,s}(A_{s,s} + A_{s,s+1}G_s)^{-1} A_{K-1,K}A_{K,K}^{-1} \end{cases} \quad (2)$$

where

$$\begin{cases} G_s = -(A_{s+1,s+1} + A_{s+1,s+2}G_{s+1})^{-1}A_{s+1,s}, \\ s = 1, 2, \cdots, K-2, \\ G_s = -(A_{s+1,s+1})^{-1}A_{s+1,s}, \ s = K-1. \end{cases}$$

The vector π_0 is calculated as the unique solution to the following system of linear algebraic equations:

$$\begin{cases} \pi_0[A_{0,0} - A_{0,1}(A_{1,1} + A_{1,2}G_1)^{-1}A_{1,0}] = 0, \\ \pi_0[e_2 + \sum_{i=1}^{c-d}(-1)^i \prod_{s=1}^{i} A_{s-1,s}(A_{s,s} + A_{s,s+1}G_s)^{-1}\mathbf{e}_2 \\ + \sum_{i=c-d+1}^{K-1}(-1)^i \sum_{s=1}^{i} A_{s-1,s}(A_{s,s} + A_{s,s+1}G_s)^{-1}\mathbf{e}_4 + \\ (-1)^K \prod_{s=1}^{K-1} A_{s-1,s}(A_{s,s} + A_{s,s+1}G_s)^{-1}A_{K-1,K}A_{K,K}^{-1}\mathbf{e}_2] = 1 \end{cases}$$

Where $\boldsymbol{e}_2$ denotes a column vector of two ones, and $\boldsymbol{e}_4$ denotes a column vector of four ones.

Proof: Expanding the matrix equation $\pi\mathbf{Q} = \mathbf{0}$, we have

$$\pi_0 A_{0,0} + \pi_1 A_{1,0} = 0, \quad (3)$$

$$\pi_i A_{i,i+1} + \pi_{i+1}A_{i+1,i+1} + \pi_{i+2}A_{i+2,i+2} = 0, \quad i = 0, 1, \cdots, K-2, \quad (4)$$

$$\pi_{K-1}A_{K-1,K} + \pi_K A_{K,K} = 0. \quad (5)$$

From Equations (4) and (5), we can obtain Equation (2). Substituting Equation (2) into Equation (3) and using the normalisation condition $\pi\mathbf{e} = 1$, we can get the probability .

IV. System Characteristics

In this section, we analyze several system characteristics based on the queue length distribution derived in section III. Let us define:

$E[L_s]$ denotes the expected number of Packet-In messages in the system (including the messages in controllers).

$E[L_q]$ indicates the expected number of Packet-In messages in the queue.

P_{loss} expresses the probability that the Packet-In messages in the system is lost.

P_{eff} means the effective Packet-In message arrival rate.

E[T] says the expected Packet-In message delay time in the system.

P{BusyCtrls=i} shows the probability that the number of busy controllers is equal to i.

E[BusyControllers] is the expected number of busy controllers.

The expressions for $E[L_s]$, $E[L_q]$, Ploss, P_{eff}, E[T], P{BusyCtrls=i} and E[BusyControllers] are given by:

$$E[L_s] = \sum_{i=1}^{c-d} i\pi_i + \sum_{i=c-d+1}^{K} i\pi_i\mathbf{e}_2,$$

$$E[L_q] = \sum_{i=c-d+1}^{K}(i-(c-d))\pi_{i,0} + \sum_{i=c+1}^{K}(i-c)\pi_{i,1},$$

$$P_{loss} = \pi_K\mathbf{e}_2,$$

$$P_{eff} = \lambda(1 - P_{loss}),$$

$$E[T] = E[L_s]/P_{eff},$$

$$P\{\text{BusyCtrls} = i\} = \begin{cases} \pi_{i,0}, \ 0 \leqslant i < c-d, \\ \sum_{i=c-d}^{K}\pi_{i,0}, \ i = c-d, \\ \pi_{i,1}, \ c-d+1 \leqslant i \leqslant c-d, \\ \sum_{i=c}^{K}\pi_{i,1}, \ i = c. \end{cases}$$

$$E[\text{BusyControllers}] = \sum_{i=1}^{c-d} i\pi_{i,0} + \sum_{i=c-d+1}^{K}(c-d)\pi_i + \sum_{i=c-d+1}^{c-1} i\pi_{i,1} + \sum_{i=c}^{K} c\pi_{i,1}.$$

V. Computer Simulation and Analysis

In order to validate the theoretical analysis, we

simulate the multi-controller scheduling policy with MATLAB simulator. We analyze the real traffic of China Education Network, and use <source address, source port, destination address, destination port, protocol> to identify a new flow and count the number of the flows. We analyze the traffic of China Education Network, the maximal number of flows per second is about 3 million, and the minimal number of flows is about 300 thousand. So in this simulation, we set the arrival rate of Packet-In messages per second range from 300 thousand to 3 million. Next, we conduct numerical studies to verify our results based on above parameters.

The expected Packet-In message delay time E[T] is displayed in Fig. 3. We fix c=200, d=20, N=400, K=600, λ=3000000 (messages/s), θ=5000 and μ varying from 20000 to 36000 (messages/s). From Fig. 3, we can note that with the increasing values of μ, the expected message delay time E[T] is a monotonically decreasing function of μ. This is because the bigger service rate μ is, the less time messages stay in the system.

The number of dormant controllers d on the expected Packet-In message delay time E[T] is demonstrated in Fig. 4. We choose c=100, N=400, K=2000, θ=2000, λ=1500000 (messages/s), μ=30000 (messages/s), and d varying from 30 to 50. From the curve of E[T], we observe that E[T] gradually growing with the growth of d. For the total service rate of the control plane is decreasing with the growth of d, then messages spend more time in the system.

This example shows the effect of Packet-In message arrive rate λ on the expected number of messages in system in Fig. 5. We pick c=160, d = 8, N=200, K=521, μ=20000, θ=5000, and λ varying from 1000000 to 2200000. We find that E[L_s] is increasing while λ is increasing. This is because the bigger the arrival rate λis, the more packets arrive at the system. Thus, more messages stay in the system under unchanged service rate μ.

The changes of service rate μ on the loss probability P_{loss} is illustrated in Fig.6. In this example, we select c=100, d=10, N=300, K=400, λ= 2000000 and θ=5000. The example performs the above parameters by varying μ from 21000 to 31000. We observe that the P_{loss}

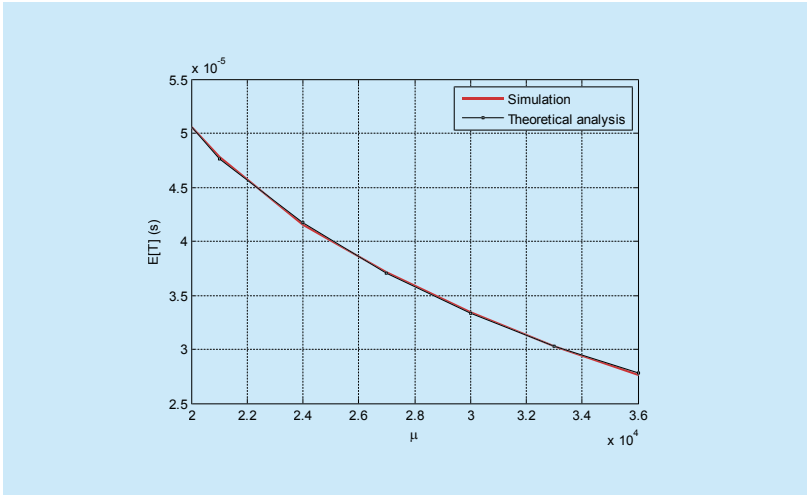

Fig.3 *The effect of μ on E[T]*

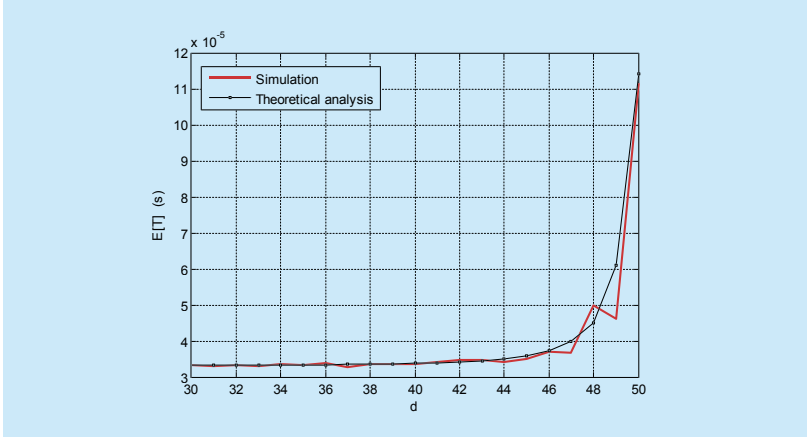

Fig.4 *The effect of d on E[T]*

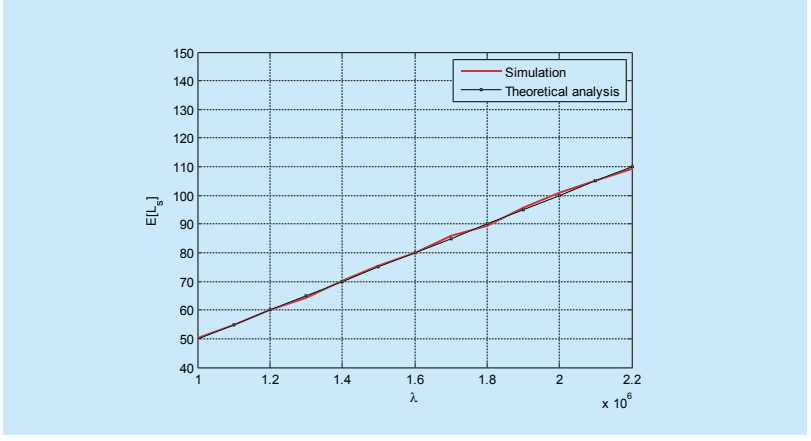

Fig.5 *The effect of λ on E[L_s]*

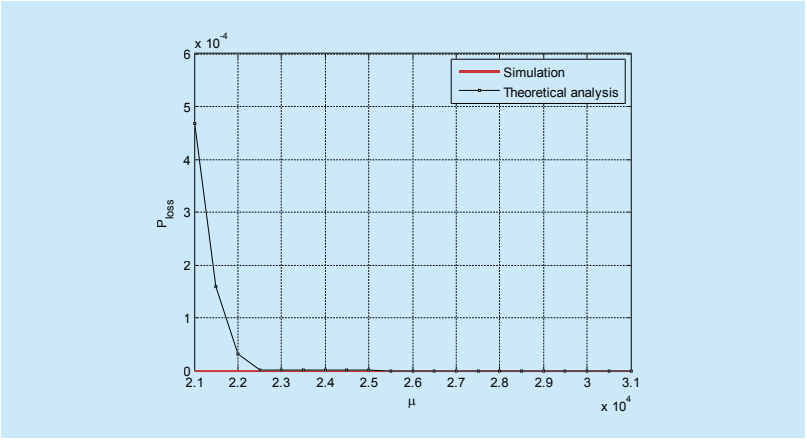

Fig.6 *The loss probability of the control plane*

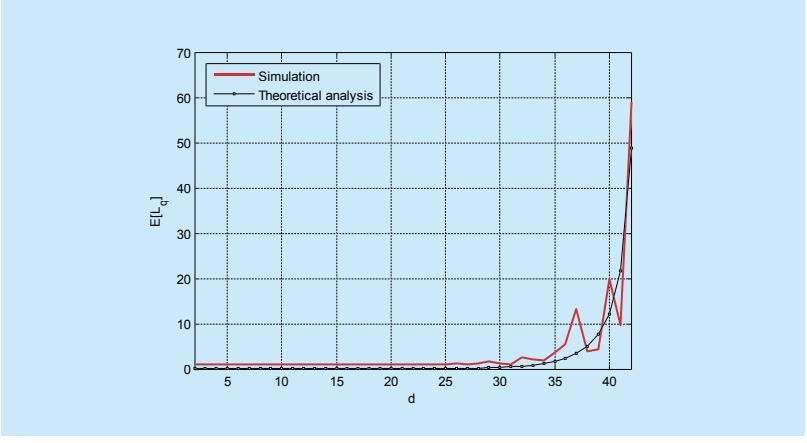

Fig.7 *The effect of d on E[L_q]*

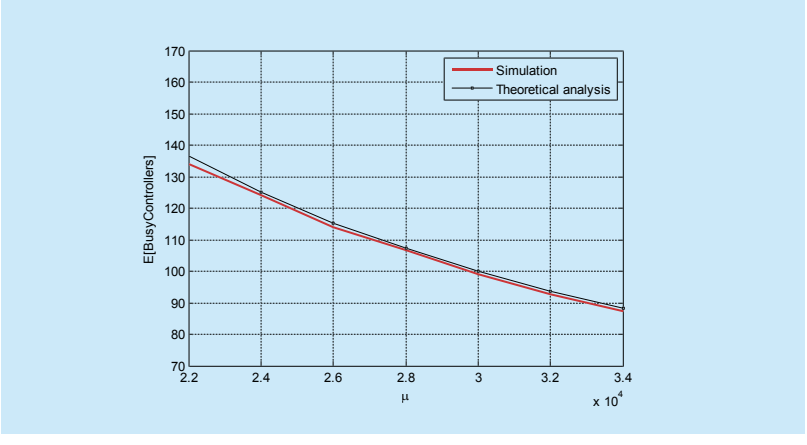

Fig. *8 The expected busy controller number*

decrease towards zero with the increasing of μ.

The number of dormant controllers *d* on the expected number of Packet-In messages in the queue E[L_q] is demonstrated in Fig. 7. Choose *c*=103, *N*= 400, *K*=2000, θ=5000, μ=50000, λ=3000000 and *d* varying from 2 to 42. From the curve of E[L_q], we observe that E[L_q] gradually growing with the growth of *d*. This is because with the increasing of *d*, the service rate of control plane is decreased, and then more messages stay in the queue.

The effect of service rate μ on the expected number of busy controllers E[BusyControllers] is shown in Fig. 8. In this example, we select *c*=160, *d* =8, *N*=300, *K*=600, λ=3000000, θ=5000, and μ varying from 20000 to 34000. We catch sight of that E[BusyControllers] is decreasing during the time that μ is increasing. This is because with the increasing of service rate μ, the leaving rate of packets is increasing and the expected number of busy controllers E[BusyControllers] is decreasing.

The probability of the number of busy controllers P{BusyCtrls=*i*} is displayed in Fig. 9. We fix parameters λ=3000000, *c*=160, *d*=10, *N* =300, *K*=600, θ = 5000, and μ varying from 30000 to 40000. From Fig. 9, we can observe that with the increasing value of μ, the peak of the curve which indicates the biggest probability controller number is decreasing. This is because the bigger the service rate μ is, the less the busy controllers are used in the system.

VI. Cost Analysis

6.1 Cost function

In this section, we develop an expected total cost function for the dormant multi-controller model. Assume the Packet-In message arrival rateand average Packet-In message service rate of single controllercan be measured in advance. The objective of the cost function is to determine the optimal designing variables, such as the total controller number c^*, the dormant controller number d^*, the state change threshold N^*, and the system capacity K^*, so as to minimize the total cost Z^*.

$$Z(c, d, N, K) = c_1 E[T] + c_2 P_{loss} + c_3 E[L_s] + c_4 (c - d) + c_5 d \quad (6)$$

In equation (6), c_1 is the cost per unit time for message delay in the control plane; c_2 is the cost per unit time for loss packet in the

control plane; c_3 is the cost per unit time for holding one message in the system; c_4 is the cost per unit time for one normal working controller; c_5 is the cost per unit time for one dormant controller.

The cost minimization problem can be presented mathematically as

$$\min Z = c_1 E[T] + c_2 P_{loss} + c_3 E[L_s] + c_4 (c-d) + c_5 d$$

$$\text{s.t. } E[\text{BusyControllers}] \le c - d,$$

$$0 < d,$$

$$d < c,$$

$$c < N,$$

$$N < K,$$

$$K < \infty.$$

6.2 Genetic algorithm

As we can see, the cost function is highly complex and nonlinear. To search the optimal values (c^*, d^*, N^*, K^*), we apply a genetic algorithm to implement the numerical searching and find the minimal cost. Genetic algorithms are global search and optimization algorithms based on biological evolution behavior proposed by Holland in 1975 [17]. The main steps of the genetic algorithm are described as follows.

6.2.1. Population initialization

In the population, chromosomes are created with four segments c, d, N, K ($d < c < N < K$) for the expected total cost function in equation (6). The length of segments can be set according to different needs. In this numerical experiment, the variables c, d, N, K are coded with 9, 9, 9 and 10 bits. Meanwhile, several parameters of the genetic algorithm are initialized, such as the population size N_p, max generation number M_G, cross over probability C_p , mutation probability M_p and the generation gap G_p.

111100000|010001100|111100000|1110000000

c | d | N | K

6.2.2. Selection

The fitness function of the genetic algorithm is the expected cost function displayed in equation (6). During each generation, all chromosomes are sorted in ascending order corresponding to their fitness function results. High fitness chromosomes of the population are eliminated, and the rest low fitness chromosomes are remained to the next generation.

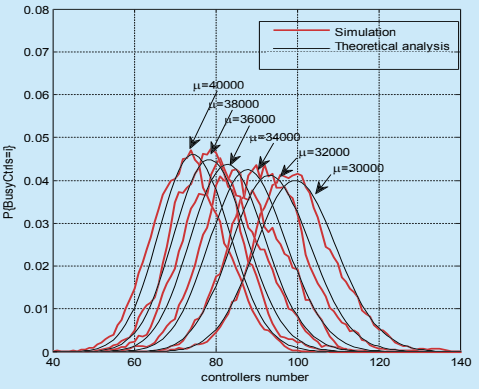

Fig.9 *The probability of P{BusyCtrls = i}*

6.2.3. Crossover

The roulette wheel method is a way used to allow the fittest individuals in generation have a greater chance of survival than other individuals. In the crossover step, roulette wheel method is implemented to pick out a good genuine father for child, and the random method is dealt with mother selection. Meanwhile, multi-point crossover means is used to combine genes of parents to produce offspring with probability C_p. To obtain competitive offspring, four points are generated randomly for crossover. Child chromosome inherits the high bits before the crossover point of every variable from its father, and inherits other low bits from its mother.

111100000|010001100|111100000|1110000000

Father

000011111|111011111|011010010|0000000111

Mother

5 | 3 | 5 | 2

Cross Point

111111111|010001111|111110010|1110000011

Child

6.2.4 Mutation

Further, multi-point mutation is applied to the

Algorithm1: find the optimal designing variables, so as to minimize the system operating cost.

```
Input: λ, μ, c1, c2, c3, c4, c5, NP , CP , GP , MP , GP, MG
Output: c*, d*, N*, K*, Z*
begin
  for i: 1 to NP
    Population_Initialization(chromosomes (i));
    Chrom_Cost(i)=fitness(chromosomes (i));
  end
  for i: 1 to MG
    % Selection
    chromosomes(i)=sort(chromosomes(i));
  % Crossover generate child
  for j: 1 to NP
    Father= RouletteWheel ();
    Mother = RandomSelect();
    Child(k)=Crossover(Father,Mother,crosspoint);
    Mututation(Child(k), MP);
  end
  for j: 1 to NP
    Child_Cost(i)=fitness(Child(i));
  end
    Reinsertion(chromosomes,Child,NP*GP );
  end
end
```

new chromosome with a low probability M_P. In the mutation step, four points are generated randomly for mutation step. The mutation bits in the chromosome are inverted.

111111111|010001111|111110010|1110000011

Chrom

1 | 3 | 2 | 4

Mutation Point

111111110|010001011|111110000|1110001011

Child

6.2.5. Reinsertion

In order to evolve towards better solutions, part of individuals will be eliminated in every generation. The number of eliminated individuals is controlled by generation gap, N_P $(1-G_P)$ individuals will be sifted out. To keep the original size of the population, N_P $(1-G_P)$ new individuals will be selected corresponding to their fitness function from the new chromosomes.

6.2.6. Next Generation

Update generation counter and population, and repeat step 2 ~ 5 until max generation M_G is met.

6.3. Algorithm

The algorithm used to find the optimal designing variables is shown in Algorithm1. In the algorithm, the time complexity of the fitness function is O(K); the roulette wheel and reinsertion functions are $O(N_P)$; the random selection and mutation functions are O(1). Then the time computation complexity of the algorithm is $O(M_G*N_P*\max(K,N_P))=O(n^3)$. Generally speaking, M_G ranges from 100 to 800, N_P ranges from 80 to 150, K ranges from 100 to 20000. So we can see that the overhead of this algorithm is low.

6.4. Numerical Results

Set the population size N_P = 100, max generation M_G =500, cross over probability C_P = 0.9, mutation probability M_P = 0.1, generation gap G_P = 0.9 and max generation M_G = 500.

Case: $c_1=1\times10^6$, $c_2=1\times10^{11}$, $c_3=30$, $c_4=100$, $c_5=50$.

In this case, we choose λ=3000000 (Packet-In messages/s), λ=1500000 (Packet-In messages/s) and λ=300000 (Packet-In messages/s) to simulate the max flow situation, the middle of the max flow and minimal flow situation, the minimal flow situation of China Education Network. Let μ varying from 20000 to 70000 (Packet-In messages/s). The numerical results are shown in Table I. From the results, we can see that if the service rate of SDN controller is given, such as μ=20000, in order to serve the maximal flow situation of data plane (λ=3000000), the total controller number c=156 can be get by the genetic algorithm. Under this situation, 6 controllers are allowed go to the dormant state. When the Packet-In message arrival rate λ decrease from 3000000 to 1500000, 74 controllers are allowed to enter the dormant state. When the Packet-In message arrival rate λ decrease to the minimum flow situation (λ=300000), 138 controllers are allowed to enter the dormant state. According to the cost function developed

in equation (6), we can get the total controller number c^*, the threshold N^*, the system capacity K^*, total cost Z^* by the genetic algorithm, and compute the dormant controller number d^* dynamically according to different Packet-In message arrival rate λ.

VII. Conclusion

This paper develops a flexible dormant model for SDN. The model allows part of idle controllers to enter the dormant state under light traffic condition for saving system operating cost. Employing queueing theory, we obtain various performance measures. Using these measures such as delay, loss probability and mean queue length, we develop an expected total cost function. Based on this function, a genetic algorithm is implemented to find the optimal values of designing variables for deployment decision making. Moreover, through computer simulation, we verify the efficiency of the dormant multi-controller model.

Acknowledgement

Supported by the National High-tech R&D Program ("863" Program) of China (No.2013AA010605), the National Science Foundation of China (No.61161140454), National Science & Technology Pillar Program of China (No.2012BAH01B01), the National Science Foundation of China (No. 61303194), the Post-Doctoral Funding of China(2013M530047), and Tsinghua-Huawei joint research project.

Table I *Searching results*

μ	λ	c^*	d^*	N^*	K^*	Z^*
20000	3000000	156	6	268	521	96894
20000	1500000	156	74	251	615	14307
20000	300000	156	138	297	661	9260.2
30000	3000000	107	5	252	438	21265
30000	1500000	107	52	196	341	9751.3
30000	300000	107	95	259	513	6358.2
40000	3000000	81	4	156	432	10874
40000	1500000	81	40	152	329	7405
40000	300000	81	70	177	404	4862.6
50000	3000000	65	3	157	347	10371
50000	1500000	65	31	156	338	5956.6
50000	300000	65	57	168	345	3885.7
60000	3000000	56	4	147	360	7476.2
60000	1500000	56	27	139	253	5082
60000	300000	56	49	155	258	3343.7
70000	3000000	50	4	147	319	6321.6
70000	1500000	50	25	103	229	4481.9
70000	300000	50	41	109	225	3093.9

References

[1] ONF White Paper. Software-Defined Networking: The New Norm for Networks [EB/OL]. 2012. https://www.opennetworking.org/.

[2] McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, Shenker S, Turner J. OpenFlow: enabling innovation in campus networks [J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2):69-74.

[3] Zheng C, Alan C L, Eugene S T. Maestro: A System for Scalable OpenFlow Control[R]. Rice University Technical Report TR10-08, 2010.

[4] Tavakoli A, Casado M, Koponen T, Shenker T. Applying NOX to the Datacenter [C]// Proceedings of the 8th ACM Workshop on Hot Topics in Networking (Hotnets): October 22-23, New York, USA. New York, NY, USA: ACM, 2009.

[5] Casado M, Freedman M, Pettit J, Luo J, McKeown N, Shenker S. Ethane: Taking control of the enterprise [C]// Proceedings of the 7th ACM SIGCOMM: August 27-31, 2007, Kyoto, Japan. New York, NY, USA: ACM, 2007.

[6] Tootoocian A, Ganjali Y. HyperFlow: A distributed control plane for OpenFlow[C] // Proceedings of 2010 Internet Network Management Workshop/ Workshop on Research on Enterprise Networking: April 27, 2010, San Jose, CA. Berkeley, USA: USENIX, 2010.

[7] Koponen T, Casado M, Gude N, Stribling J, Poutievski L, Zhu M, Ramanathan R, Iwata Y, Inoue H, Hama T, Shenker S. Onix: a distributed control platform for large-scale production networks[C] // Proceedings of the 9th USENIX conference on Operating systems design and implementation: October 4-6, 2010, Vancouver, Canada. Berkeley, USA : USENIX, 2010.

[8] Yu M, Rexford J, Freedman J M, Wang J. Scalable Flow-Based Networking with DIFANE [J]. ACM SIGCOMM Computer Communication Review, 2010, 40(4): 351-362.

[9] Yeganeh H S, Ganjali Y. Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications [C]// Proceedings of the Hot Topics in Software Defined Networking (HotSDN): August 13-17, 2012, Helsinki, Finland. New York, NY, USA: ACM, 2012.

[10] Dixit A, Hao F, Mukherjee S, Lakshman T V , Kompella R. Towards an Elastic Distributed SDN Controller. // Proceedings of the Hot Topics in Software Defined Networking (HotSDN): August 16, 2013, Hong Kong, China. New York, NY, USA: ACM, 2013.
[11] Luo M, Tian Y, Li Q, Wang J and Chou W. SOX - A Generalized and Extensible Smart Network Openflow Controller[C] // Proceedings of the first SDN & OpenFlow World Congress: October 22-24, 2012, Darmstadt, Germany. California, CA, USA: ONF, 2012.
[12] Yadin M, Naor P. Queueing systems with a removable service station [J]. Operational Research Quarterly, 1963, 14(4):393-405.
[13] Heyman D. Optimal operating systems in M/G/1 queueing systems [J]. Operational Research, 1968, 16: 362-382.
[14] Shen L M, Jin S F, Tian N S. The M/M/c Queue with N policy and Multiple Vacation of Partial Servers [J]. Journal of Engineering Mathmatics, 2004, 21(2):238-244.
[15] Ke J C, Lin H C, Yang J Y, Zhang Z G. Optimal (d, c) vacation policy for a finite buffer M/M/c queue with unreliable servers and repairs [J]. Applied Mathematical Modelling, 2009, 33(10):3949-3962.
[16] Kim C, Dudin S, Klimenok V. The MAP/PH/1/N queue with flows of customers as a model for traffic control in telecommunication networks [J]. Performance Evaluation, 2009, 66(9-10):564-579.
[17] J. Holland. Adaptation in Natural and Artificial Systems[M]. The University of Michigan Press, Ann Arbor, 1975.

Biographies

FU Yonghong, Ph.D. candidate with Department of Computer Science, Tsinghua University, China. Her main research interests include network architecture and software Defined Networking. Email: fuyh11@mails.tsinghua.edu.cn

BI Jun, Professor and Ph.D. supervisor with Institute for Network Sciences and Cyberspace, Tsinghua University, China. His main research interests include Internet architecture and protocols, future Internet (SDN and NDN), Internet routing, and source address validation and traceback. *The corresponding author. Email: junbi@tsinghua.edu.cn

WU Jianping, Professor and Ph.D. supervisor with Institute for Network Sciences and Cyberspace, Tsinghua University, China. His main research interests include Internet architecture and protocols, IPv4/IPv6 transition, Internet routing, and source address validation. Email: jianping@cernet.edu.cn

CHEN Ze, Undergraduate student with Department of Computer Science, Tsinghua University, China. His main research interests include network architecture and software defined networking. Email: chen-z10@mails.tsinghua.edu.cn

WANG Ke, Undergraduate student with Department of Electronic Engineering, Tsinghua University, China. His main research interests include network architecture and software defined networking. Email: wang-k10@mails.tsinghua.edu.cn

LUO Min, Head and Chief Architect, Shannon Lab, Huawei Technologies Company, USA. His main research interests include network architecture and software defined networking, enterprise information systems, cloud computing. Email: min.ch.luo@huawei.com