AAA: High Agile Adaptive Flow-Awareness Network for SDN

He Cai[1], Jun Deng[1], Xiaofei Wang[1]
[1]Tianjin Key Laboratory of Advanced Networking, Tianjin University, Tianjin, China.

***Abstract*—The abstract goes here.**

1. Introduction

With the data traffic and network scale rapidly increaing, there exists huge demand for scalable network management. Meanwhile, network monitoring and application awareness play a increasingly critical role in Quality of Service(Qos),Traffic Engineering(TE) and cyber security. Briefly, application awareness is a basic technology to enhance automation and intelligence of the network. It is divided into two processes: packet acquisition and traffic identification. Packet acquisition refers to capturing packets from switches through a mechanism or an algorithm. Traffic identification refers to parsing the five-tuple information of packets from different layer according to OSI model ,then recognizing the application layer protocol with the help of DPI tools. Application-aware network can improve the visibility of itself , promote integration of different business and eliminate faults quickly .However, the application awareness need integrate the high precision,high efficiency with real time, which is still a challange owing to the volume and variety of data in the large-scale network.

Software-defined network (SDN) is a new technical architecture which decouple the network control plane from the data-forwarding plane. It advocates building an open and programmable network to provide flexible, central controlled(or centralized) and globally visible network services, through which SDN can facilitate the operation and maintenance of the data center(DC) network. In a software-defined network, packet acquisition depends on OpenFlow(OF) protocol,which is varied from the Netflow and Sflow used in traditional networks.

Based on port, payload, and traffic behavior characteristics, DPI can identify a variety of information including the application layer protocol of a data flow, and be applied in application-aware network. In traditional networks, DPI devices are bound to the data plane, which makes it impossible to visualize global fine-grained traffic in real time. Therefore,many people are concerned about the research and optimization of the combination of SDN and application awareness. However, most of the current solutions are to deploy DPI in the SDN controller.In this case, parsing each single packet will be computationally heavy for controller. In addition, network scale, number of sampling nodes, sampling frequency and repetition rate of packet all increase performance consumption of controllers.On the other hand, in order to improve the accuracy of application recognition, the system must be able to capture continuous packets of the same flow regarding to the characteristics of DPI. To solve the above problems, a agile, adaptive and cooperative sampling mechanism which can be applied to large-scale data center network is urgently needed.

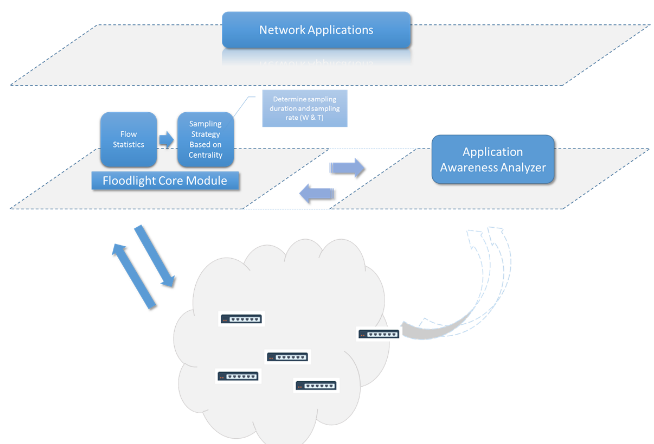


Figure 1: The System Architecture Of AAA

2. Related Work

- A detailed introduction to the background: Why build an application-aware network - the contrast between traditional and SDN networks.
- Introducting the problems existing in building a large-scale application-aware network (a reasonable collection of traffic in a large-scale network environment is a basic problem in the field of traffic monitoring and traffic engineering).
- And including the solutions already on the basic problem, and the various problems that exist.
- Introducting the Intermediary centrality algorithm.
- Outline the algorithms and strategies we use.

3. System Model and Design

3.1. Problem Model

We start with the four aspects of adaptability, agility, accuracy, efficiency, and synergy to abstract and model the problem. Build a AAA Co-Sampling Model. We abstract this problem with a new perspective, making the problem more intuitive and modeling the problem.

We believe that there is a cooperative relationship between the sampling between nodes. This relationship is multi-dimensional, which is reflected in the mutual constraint relationship between them, the coverage relationship of the current network, the static relationship in the topology, and so on.

Fig. 1 shows our intuitive abstraction of the problem. For a sampling period T, the stream set F in the entire network is like an area depicted by a red outline, and for R1..Rn, each node is covered with 0 or more streams, which is recorded as a set F_i^c, as shown in the flow information matrix, is a shaded area in gray in Fig. 1. The red dotted area is the newly arrived stream that may be covered by the period Ri corresponding to the period Ri. The overlapping area between the gray areas is the overlapping part of the flow covered between the nodes, $F_i^c \bigcap F_j^c$. The overlapping portion between the red dotted areas is the overlapping portion of the newly arrived flow covered between the nodes. Some nodes contain both gray and red areas; while some nodes only contain red or gray areas, which represents the value of each node in the period. The larger the area covered, the more streams the node covers. Therefore, the Flow-Level sampling problem can be intuitively converted to an area coverage maximization problem. That is, under the given collector processing power and other constraints, the sampling time allocation of each node is realized, so that the coverage area is maximized (the maximum number of stream coverage).

Under the definition of this problem, we propose a quantization model based on Slot partition time. Fig. 2 shows the way this model is used. First we divide the sampling period T into L equal length Slots; for each Ri, we need to determine its sampling Slot set $\widetilde{S}_i$. For any switch, the value it can generate in a Slot is expressed as v_i, which is visually represented as the number of streams sampled. For all nodes, the number and order of the Slots they are assigned to are calculated. The reason is that the number of Slots is positively correlated with the value generated by a node, and the more time is allocated to a node, The expected value of sampling to the stream is larger, and the order of the Slot of each node implicitly expresses the mutual constraint between the nodes: because the two nodes have overlapping flows, and when the coverage area is maximized, it should be removed. Multiple accumulations of overlapping areas between nodes in the same Slot. For the value v_i produced by R_i, the quantization should take into account the expected number of known flows in a unit Slot time, and should also consider the number of new flows that may arrive within the period T, but For the unknown stream, even if we quantify the number of arrivals of a switch through the arrival distribution model of some flows, we cannot know the relationship between the unknown flows arriving between the nodes, and the paths and nodes of the unknown flow cannot be predicted. The relationship between the overlapping flows is so difficult to quantify the constraint relationship between the newly arrived flows to the nodes in the period T. However, for a node, its static Topology influence and short-term activity can be used to approximate its value in the period T. They represent one of the nodes in the week T in the whole node. Influence, the greater the influence, the more likely it is to cover more flows. Moreover, these two attributes are private to the node, and there is no constraint relationship with other nodes. The number of known flows/the total number of streams covered by the nodes in the unit slot is the coverage of the current stream brought by the nodes in the unit time Slot, which we call the dynamic influence of the flow.

We give the following optimization model, Maximize Influence (formula), which is a transformation based on the traffic coverage problem: convert the number of coverage of the stream into coverage to quantify the node in unit t Dynamic influence within; use Si and Hi to quantify the influence of nodes within T. Therefore, in t unit time, the influence of a certain node should be considered in total and we use weighted way to get the comprehensive influence value of the node. In the quantification of D_i, we assume that the arrival strength of the stream f_i packet obeys the Poisson distribution with intensity λ_i.

$$\begin{aligned} \max \sum_{i}^{n} (\alpha \cdot \frac{\delta(v_i, \left|\widetilde{S}_i\right|)}{|F^c|} + \frac{\left|\widetilde{S}_i\right|}{T/t} \cdot (\beta \cdot S_i + \gamma \cdot H_i)) \\ - \frac{\alpha}{|F^c|} \cdot \sum_{f_k \in F^c} \sum_{l=1}^{T/t} \left(P\left\{N_p^k(t) > 0\right\} \cdot \psi\left(f_k, s^l\right) \right) \end{aligned} \tag{1}$$

subject to:

$$v_i = \sum_{f_k \in F^C} P\left\{N_p^k(t) > 0\right\} \tag{2}$$

$$\delta\left(v_i, \left|\widetilde{S}_i\right|\right) = \begin{cases} v_i \cdot \left|\widetilde{S}_i\right|, v_i \cdot \left|\widetilde{S}_i\right| < |F_i^c| \\ |F_i^c| \quad , \text{ ELSE} \end{cases} \tag{3}$$

$$U = \left\{R_i, f_k \in F_i^c \wedge s^l \in \widetilde{S}_i\right\} \tag{4}$$

$$\psi\left(f_k, s^l\right) = \begin{cases} |U| - 1, |U| \geq 1 \\ 0 \quad\quad , |U| = 0 \end{cases} \tag{5}$$

$$\alpha + \beta + \gamma = 1 \tag{6}$$

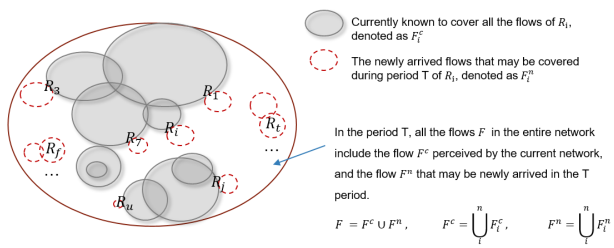
$$\sum\nolimits_i^n f(\widetilde{S}_i) \leq K, \quad f(\widetilde{S}_i) = \begin{cases} 1, & \left|\widetilde{S}_i\right| \geq 1 \\ 0, & \left|\widetilde{S}_i\right| = 0 \end{cases} \tag{7}$$

$$\sum\nolimits_i^n w_i \cdot \varphi(\widetilde{S}_i, s^l) \leq C{\cdot}t, \forall \mathrm{s}^l \wedge \varphi(\widetilde{S}_i, s^l) = \begin{cases} 0, s^l \notin \widetilde{S}_i \\ t, s^l \in \widetilde{S}_i \end{cases} \tag{8}$$

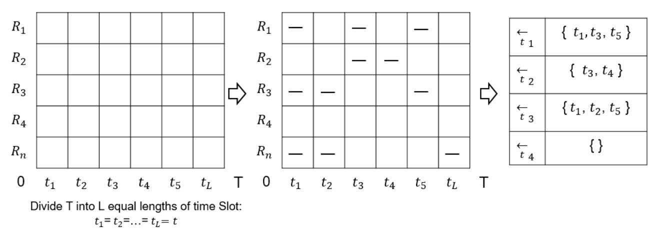
$$l \leq {}^{T}\!/_{t}, \forall s^l \wedge t \leq T \tag{9}$$

$$K \leq n, \forall R_i \tag{10}$$

$$\left|\widetilde{S}_i\right| \leq {}^{T}\!/_{t}, \forall i \tag{11}$$



(a) area coverage



(b) time slot allocation

Figure 2: overview of model

For the model of maximizing impact problems, the coverage of known traffic, the coverage of unknown new flows, the selection of nodes, the allocation of time slots, and the scheduling of Slot time series are fully considered. In the scheduling of time Slot, because the overlapping flows between nodes will cause conflicts under the same Slot, the overall coverage is reduced, so in order to get the optimal solution, the more overlapping the two nodes, if they are both When more than 0 Slots are allocated, the number of Slots they overlap should be as small as possible, which in turn solves another important sampling problem: how to reduce the problem of duplicate packets. Because each node contains the same stream, in order to optimize the results, it will be mutually exclusive on the Slot. Therefore, in the optimized solution, the sequence of $\widetilde{S}_i$ of each node can make the area covered by overlapping sampling in the whole cycle. Minimized, thus greatly reducing the repetition rate of the package while ensuring maximum sampling accuracy. Therefore, the nodes are cooperative, and through the constraint relationship between each other, the flow coverage relationship, the static topology relationship, the activity degree, etc., and finally maximize the number of sampled streams. (In the sampling process, when multiple nodes are sampled, a large number of duplicate packets are generated, because the stream may pass through any number of nodes, and the same packet will be collected at multiple nodes, which not only wastes valuable sampling resources, At the same time, the sampling accuracy is reduced. The repeated packets are reduced as much as possible, and the problem is avoided by the superposition relationship between the nodes, which not only improves the sampling accuracy, but also improves the efficiency of the upper application of the collector).

The model contains a number of sub-constrained sub-problems. We consider the complexity and feasibility while quantifying, and it is actually difficult to directly solve the solution to the problem. Therefore, we decompose the problem model into three sub-models: node selection, Slot number allocation, and Slot sequence arrangement. For each sub-problem, we model it independently and use an independent algorithm to solve the optimal solution or approximate optimal solution. In the end, the approximation of the problem can be effectively obtained. This problem is not only a variant of multiple backpacks, but also adds enough constraints on the issue of multiple backpacks. Asking to solve this problem is an NP-hard problem. Therefore, we break it down into three parts to approximate the problem.

TABLE 1: table

Nonation	Explanation
M	the current flow information matrix
S	selected switches set
sw_j	the j-th switch
f_i	the i-th flow
m_{ij}	the each value for f_i and sw_j in M,either 1 or 0
c_j	the betweenness centrality of sw_j
c_{max}	the max betweenness centrality of sw_j
I	the current number of flows
J	the current number of switches

3.2. Sampling Point Selection

According to the definition of model (1), select the K most influential nodes as sampling nodes and assign Slots. In the optimization model (1), selecting the sampling node and assigning the number of Slots and determining the Slot order of each sampling node. The constraint between them is the overlapping problem of known flows between the nodes, so the three problems should be separated separately. First, the problem of overlapping of streams between nodes needs to be removed. We use the node influence power mode in model (1) to comprehensively quantify each node and select the top K influence maximization nodes through dynamic influence, static topological influence, and node historical influence. In the process of selecting the round (a total of K rounds), the high-impact nodes are used to privatize the overlapping flows to solve the overlapping problem of the flow. At this time, the dynamic influence force Di indicates the influence of the node based on the intermediate degree of the flow. Dik indicates the dynamic influence of node i when the kth node is selected during the Kth round selection. In the Kth round selection, if the i node has the greatest comprehensive influence, it will privatize all the flows it contains but Does not contain the stream contained in the nodes in the 1-k-1 round. Formula (15) gives a quantitative formula for the comprehensive influence.

The principle of the algorithm will be stated next, with respect to the notations in Table I being used throughout the paper.

Firstly, initialize the matrix M=[m_{ij}] and the betweenness centrality c_j. Fig.3-a shows a subnet topology, where there are 6 switch nodes and 6 flows. And we define: if f_i passes through sw_j, the m_{ij}=1, otherwise m_{ij}=0. After initialization,as Fig.3-b shows,we get a $I * J$ two-dimension matrix. Then calculate c_j and c_{max}.

$$c_{max} = \max\{c_j \mid c_j = \sum_i m_{ij}, i \in [1, I], j \in [1, J] \wedge i, j \in Z\} \tag{12}$$

$$D_i^k = \left| F_i - \bigcup_c^{k-1} \widetilde{F_c} \right| / \left| F - \bigcup_c^{k-1} \widetilde{F_c} \right| \tag{13}$$

$$H_i = TF_i / TF \tag{14}$$

$$S_i = C_i / \sum_i^n C_i \tag{15}$$

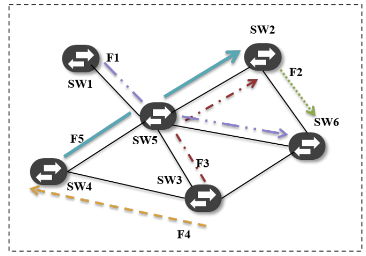
$$I_i^k = \alpha \cdot D_i^k + \beta \cdot S_i + \gamma \cdot H_i \tag{16}$$

Senondly, elect the node with highest betweenness centrality as the sampling node and change m_{ij} until each m_{ij} = 0. As shown in Fig.4, c_{max} = c_3. Hence,the sw_3 is the first sampling node. Owing to f_1,f_2,f_4,f_6 pass through the sw_3, make m_{ij} = 0(i=1,2,4,6,$j \in [1, J]$). Then we can get a new marrix M and calculate new c_j and c_{max} used for the next election. Repeating the above method, and electing the sampling node sw_4. Finally,we get S=sw_3,sw_4,when each m_{ij}=0.

Algorithm 1 Sampling Point Selection

Input:
The set of routers: R
The size of node will be selected: K
The current flow information matrix: M

```
define R^s = {} // The Set of Selected Routers
for k = 1; k < K; k++ do
    for each R_i ∈ R − R^s do
        if I_i^k > max then
            max = I_i^k
            SR = R_i
        end if
    end for
    put SR to R^s
    mark SR as R_k in R^s
end for
return R^s
```



	S1	S2	S3	S4	S5	S6
F1	1	0	0	0	1	1
F2	0	1	0	0	0	1
F3	0	1	1	0	1	0
F4	0	0	1	1	0	0
F5	0	1	0	1	1	0
Betweenness Centrality	1	3	2	2	3	2

Figure 3: Intermediary center based on the number of streams

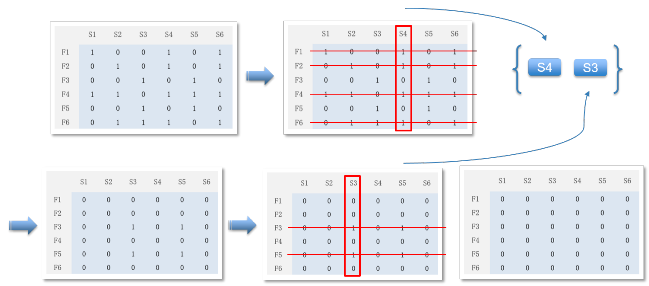


Figure 4: Sampling Point Selection

3.3. Allocation of Time Slot

After the nodes are selected, each node needs to be assigned a number of slots. This is a typical multi-clip problem: there are a total of K items to choose $(R_1^s - R_k^s)$, each type of item to be selected has $\frac{T}{t}$ ($\frac{T}{t}$ Slots can be selected), and each item is in one unit Slot t The benefit is $v(R_i, t_i)$ and the cost is $W(R_i, t_i)$. Therefore, we can use the algorithm for solving multiple backpacks to allocate the number of slots for each Ri. $v(R_i, t_i)$ is the part of Model(l) in the model , take $|\widetilde{S}|$ to 1. And $W(R_i, t_i)$ = the packet rate pkts/t of the unit t time Ri. When quantifying Di of $v(R_i, t_i)$, we assume in model (1) that the arrival rate of the packet of fi obeys the Poisson distribution. In the actual process, it is difficult to know the λ intensity of the packet of a stream. We can pass fi The rate of the package to characterize. When using multiple backpacks to solve this problem, there may be a phenomenon of hunger distribution: some low-impact nodes are assigned to 0 slots. This phenomenon is not avoided. We assign a Slot to each node from the high-impact to low-impact nodes (Rs1-Rsk) in advance (in the case where the cost does not exceed the total constraint), and then use multiple backpacks to solve You can try to avoid hunger.

When solving multiple backpacks, in addition to the problem of hunger, the solution efficiency is the complexity of $O(n^2)$. When T is too large or t is too small and C is large, it is unacceptable to solve the optimal solution time. We give a simple and efficient strategy. Algorithm 2 shows this approach.

3.4. Order of Time Slot

Determining time series for each node, in Model(1), reflects the effects of overlapping relationships between nodes, such as the accuracy of the sampling of the system and the repetition rate of the sample packets. In the second chapter, we use the high-impact privatization of overlapping flows to approximate the effect of the precision caused by overlapping flows between nodes. Therefore, in this section,

Algorithm 2 Allocation of Time Slot Based on XXX

Input: K ,L ,C , weight $W = \{w_1, w_2..., w_k\}$,value $V = \{v_1, v_2..., v_k\}$

while $C > 0$ **do**
  $Cas = 1$
  for $k = 1$; $k < K$; $k++$ **do**
    if $C >= w_i * Cas$ **then**
      $count[i] += Cas$
      $C = C - w_i * Cas$
    end if
  end for
end while
return reverse iteration,output $N = \{n_1, n_2..., n_k\}$

we reduce the repetition rate of the packet by optimizing the sampling sequence of each node Slot.

This optimization problem can be defined by the following formula(16). Where $\left|\widetilde{S_i} \bigcap \widetilde{S_j}\right|$ represents the number of overlaps of the time slots of the two nodes, and $\left|\widetilde{F_i} \bigcap \widetilde{F_j}\right|$ represents the number of overlapping flows between the two nodes of ij. Therefore, this formula embodies the overlap area of the flow in the entire sampling system.

$$\min \sum (\left|\widetilde{S_i} \bigcap \widetilde{S_j}\right| \cdot \left|\widetilde{F_i} \bigcap \widetilde{F_j}\right|), \forall i, j \wedge i \neq j \tag{17}$$

From the definition of the problem, the search backtracking method can be used to solve the optimal solution, but it is a problem that cannot be solved in a polynomial time. Therefore, we consider a simple greedy algorithm to solve the approximate optimal solution of the problem.

Algorithm 3 Order of Time Slot Based on Greedy

Input: M , S , c_j

while $CNT[i] > 0, \exists i \wedge i = 1, 2..., k$ **do**
  for $i = 1$; $i <= K$; $i++$ **do**
    if $CNT[i] > 0$ **then**
      $Min = Max\ Integer$
      for $l = 1$; $l < \frac{T}{t}$; $l++$ **do**
        if $M^{slot}[i][l] = 0$ **then**
          $temp = \sum_{j=1 \wedge j!=i}^{K} (\left|\widetilde{F_i} \bigcap \widetilde{F_j}\right| \cdot M^{slot}[j][l]) + H[j]$
          if $temp < Min$ **then**
            $Min = temp$; $Sp = l$
          end if
        end if
      end for
      $M^{slot}[i][Sp] = 1$; $CNT[i]--$; $H[Sp] = Min$
    end if
  end for
end while
return M^{slot}

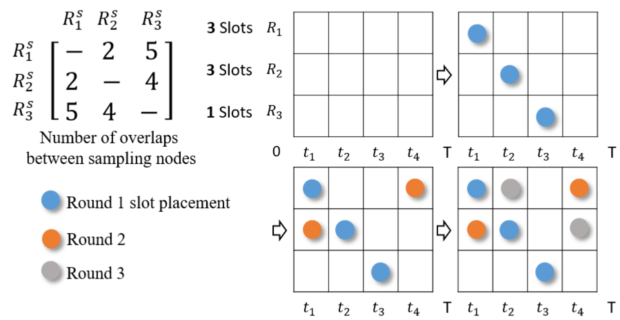
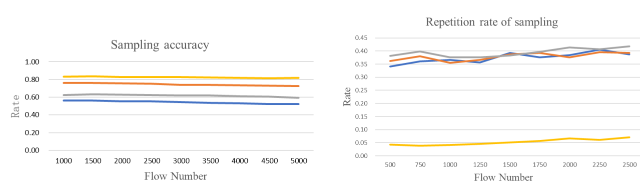


Figure 5: comparison of sampling flow number at different slot

4. Experiments and results

In order to verify the effectiveness and performance of our algorithm, we have built a laboratory bed based on floodlight controller and openvswitch + mininet. The whole experimental bed contains 12 Dell XPS hosts, 20 core CPU, and Ubuntu 16.06.2 LTS. One runs a floodlight controller that fuses our algorithm, the other runs a data collector, and the remaining 10 deploy a network topology with 110 switch nodes and 50 host nodes. The experimental traffic dataset comes from the open project "the WIDE Project". We selected data from 14:00-14:15 in August 6, 2018. After cleaning and screening, we collate 5000 data streams for experiments. In the experiment, the number of data streams changed from 1000 to 5000.. We implement four algorithms: Random-K, top-K based on the extended median centrality, top-K based on the standard median centrality, our algorithm XXX. Based on the above four algorithms, we have made a comparative experiment in three measurement mechanisms: sampling accuracy, packet repetition rate and the number of rat streams collected. Fig.x shows the comparison of sampling accuracy in different algorithms. Our algorithm is 7% higher than Top-k and over 20% than the other two algorithms. From Fig.x, we can see that different algorithms do almost the same amount of elephant flow collection. In fact, our algorithm only takes more part of the rat flow than other algorithms. And Fig.x shows that our algorithm is effective in reducing duplication and reducing it by more than 30%.



(a) comparison of sampling accuracy (b) comparison of repetition rate

Figure 6: comparison with respect to different algorithms

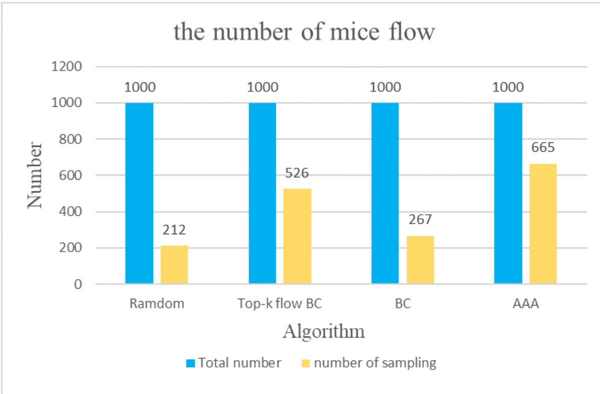


Figure 7: comparison of different algorithms in number of mice flow

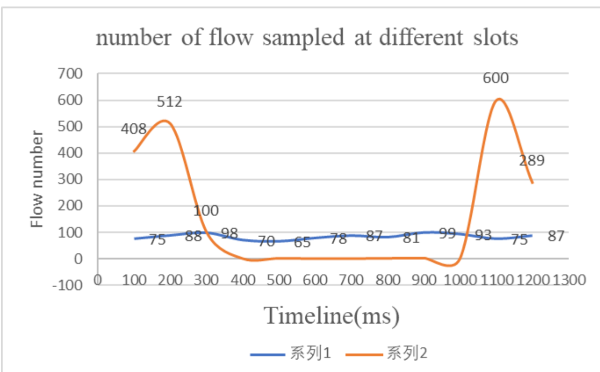


Figure 8: comparison of sampling flow number at different slot

5. Conclusion

- Lab environment
- Sampling accuracy comparison
- Sampling repetition rate comparison
- Greedy centrality algorithm experimental results
- Deduplication rate algorithm comparison
- Experimental comparison of adaptive co-sampling algorithm

References

[1] S. Yoon, T. Ha, S. Kim and H. Lim, Scalable Traffic Sampling using Centrality Measure on Software-Defined Networks, in *IEEE Communications Magazine*, pp.43-49, July 2017.

[2] M. Malboubi, L. Wang, C.N. Chuah, P. Sharma, Intelligent SDN based Traffic (de)Aggregation and Measurement Paradigm (iSTAMP), in *IEEE INFOCOM*, Apr 2014.

[3] L. Tong and W. Gao, Application-Aware Traffic Scheduling for Workload Offloading in Mobile Clouds, in *IEEE INFOCOM*, pp.1-9, Apr 2016.

[4] J. Jiang, S. Ma, B. Li and B. Li, Symbiosis: Network-Aware Task Scheduling in Data-Parallel Frameworks, in *IEEE INFOCOM*, pp.10-14, Apr 2016.

[5] P. Bakopoulos, K. Christodoulopoulos, G. Landi et al, NEPHELE: An End-to-End Scalable and Dynamically Reconfgurable Optical Architecture for Application-Aware SDN Cloud Data Centers, in *IEEE Communitions Magazine*, pp.178-188, Feb 2018.

[6] J. Xu, J.Y. Wang, Q. Qi, H.F. Sun and B. He, IARA: An Intelligent Application-aware VNF for Network Resource Allocation with Deep Learning, in *IEEE SECON*, pp.1-3, June 2018.

[7] J. Suh, T.T. Kwon, C. Dixon, W. Felter and J. Carter, OpenSample: A Low-latency, Sampling-based Measurement Platform for Commodity SDN, in *IEEE ICDCS*, pp.228-237, July 2014.

[8] Z. Su, T. Wang, Y. Xia and M. Hamdi, CeMon: A Cost-effective Flow Monitoring System in Software Defined Networks, in *Computer Networks*, pp.101-115, Dec 2015.

[9] N.F. Huang, C.C. Li, C.H. Li, C.C. Chen,C.H. C and I.H. Hsu, Application Identification System or SDN QoS based on Machine Learning and DNS Responses, in *APNOMS*, pp.407-410, Sept 2017.

[10] S. Jeong, D. Lee, J. Hyun, J. Li, and J.W. Hong, Application-aware Traffic Engineering in Software-Defined Network, in *APNOMS*, pp.315-318, Sept 2017.

[11] G. Cheng and Y. Tang, eOpenFlow: Software Defined Sampling via a Highly Adoptable OpenFlow Extension, in *IEEE ICC*, pp.1-6, May 2017.

[12] S. Zhao and D. Medhi, Application Performance Optimization Using Application-Aware Networking, in *IEEE NOMS*,pp.1-6, Apr 2018.

[13] M. Malboubi, S.M. Peng, P. Sharma and C.N. Chuah, A Learning-based Measurement Framework for Traffic Matrix Inference in Software Defined Networks, in *Computers & Electrical Engineering*, Dec 2017.

[14] K. Bilal, S.U. Khan, L. Zhang et al, Quantitative comparisons of the state-of-the-art data center architectures, in *Concurrency & Computation Practice & Experience*, Dec 2017.