

AAA: High Agile Adaptive Flow-Awareness Network for SDN

He Cai¹, Jun Deng¹, Xiaofei Wang¹

¹Tianjin Key Laboratory of Advanced Networking, Tianjin University, Tianjin, China.

Abstract—The abstract goes here.

1. Introduction

With the data traffic and network scale rapidly increasing, there exists huge demand for scalable network management. Meanwhile, network monitoring and application awareness play an 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 challenge 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.

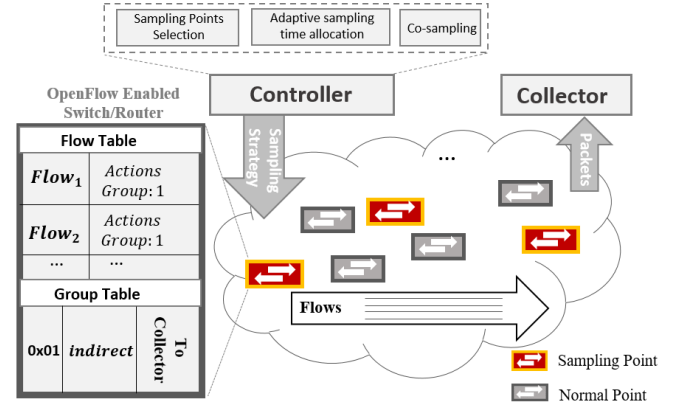


Figure 1: The System Architecture Of AAA

2. Related Work

- A detailed introduction to the background: Why build an application-aware network - the contrast between traditional and SDN networks.
- Introducing 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.
- Introducing the Intermediary centrality algorithm.
- Outline the algorithms and strategies we use.

3. SYSTEM DESCRIPTION AND PROBLEM STATEMENT

Figure 1 shows the framework of AAA. In the AAA mode, the SPS mode is used to collect the flow table entries

of each group. All packets representing all the flows passing through the switch are copied to the unified group table entry to perform the actions defined by the group entry. When the controller controls the initialization of the group of entries, the action is initialized to point to the collector or discarded. Therefore, when the controller needs to control a switch to sample or stop sampling, it only needs to simply send a Group Mod message to the corresponding switch. When the action is Drop, the sampling is stopped. When it is directed to the exit of the collector, the sampling is started. . This not only makes use of the pure OpenFlow protocol, but also controls it more precisely based on the controller's global. Assuming that the sampling period is T , through the adaptive coordination algorithm of AAA, the sampling points are selected and the sampling duration is allocated for them, and the sampling order is determined and the strategy is decentralized to the corresponding switch to make them cooperate sampling.

3.1. IMM Model

Sampling in a large-scale network requires a maximum challenge to maximize the flow-level sampling accuracy and increase the sampling efficiency ratio while satisfying the low intrusion of the network and the maximum limit of the Collector (IDS eg) analysis capability. . Therefore, highly agile and adaptive algorithms are needed to set the sampling strategy based on the real-time conditions of the current network. When the network size is large, assuming that the number of switches is n , K ($K \leq n$) nodes are generally selected for sampling, and K is determined according to actual conditions. In the sampling in each period T , if the capacity of the collector is C packets/ T , then: Total Sampling Packet in $T \leq C$. In the sampling process, under the premise of given C and K , the reasons that affect the sampling accuracy include the selection of sampling nodes and the allocation of sampling time for each node. Another reason why the sampling accuracy is affected should be the sampling ratio (sampled non-repetitive/total sampling). Repeated packet acquisition not only occupies limited sampling resources, but also limits the sampling accuracy. Reduce the efficiency of the upper application (IDS eg). The repetition rate is generated because multiple switching nodes cover the same stream and are collected at almost the same time. Each sampling node should have reasonable coordination and cooperation to reduce the repetition rate and improve the sampling accuracy.

The most reasonable node selection and time allocation, the optimal co-sampling strategy to maximize the flow-level sampling accuracy. Therefore, we consider from these three perspectives to model the maximum flow-level sampling accuracy. First we propose an intuitive perspective based on maximizing area coverage to analyze the problem. Figure 2 shows this idea: In a sampling period T , the set of flows F in the entire network is like an area depicted by a solid red line. $R_1..R_n$, each node is covered by 0 or more streams, the set F_i^c , which is a shaded area in gray in Fig. 2, which represents the value of the node for the coverage

of the known stream in the whole network. recorded as dynamic value. The red dashed area, which is the newly arrived stream that node R_i may cover in T , represents the potential value of the node in T time: these streams arrive after the sampling strategy is decentralized and are therefore not perceived by the sampling algorithm . The overlapping area between the gray areas is the overlapping part of the flow covered between the nodes, $F_i^c \cap F_j^c$. 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, The greater the number of streams detected by the node, the greater the value of the node. Therefore, the Flow-Level sampling problem can be intuitively converted into 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).

On the problem of maximizing area coverage, for the same stream, if there is temporal overlap in sampling on different nodes, the overlap should be subtracted when calculating the overall coverage area. However, the red region of each node is unknown to the sampling strategy. Even if the distribution model of the flow can be analyzed, the overlapping relationship of these unknown flows at each node cannot be known. Therefore the potential value of a node requires an independent (independent of other nodes) quantization. For a node, its intermediateity in the Topology [2] and the proportion of its historical flow can be used to evaluate the potential impact of the node, and these influences represent the potential value of the node, namely: potential impact The greater the force, the greater the value that the node may create in a unit of time. The value generated by the node during unit time t should be equal to the dynamic value + potential value. Assume that the arrival of the packet of the stream f_i obeys the Poisson distribution of da , so if any switch captures at least one packet of f_i , then the stream is considered to be successfully captured, f_i The probability of being captured in unit time t is $PN_p^k(t) > 0$. For R_i , if it is assigned a t , the dynamic value brought by the node (actually the number of expected to capture the known stream) is λ , then its area coverage is $D_i/|F^c|$, the quantization method is based on an extended version of [2] The mediation center metric, which we call the dynamic impact of R_i over the t -sampling duration. The OSPF-based mediation in Topology measures the influence of nodes in the topology. We use a standardized intermediate degree [2], and the influence of R_i in the topology over the entire sampling period T is recorded as S_i . If the higher the S_i , the node is likely to go through more streams. If at some point in the network, the dynamic influence of R_i is greater than R_j , that is, the current R_i has more streams, but the dynamic impact of R_j is higher than R_i , so in the following time, R_j has a greater probability of going through more streams. The historical flow ratio of node R_i reflects the activity of the node throughout the network life cycle. We call it the historical influence of R_i over the entire sampling period T , denoted as H_i , $H_i = TF_i/TF$. The potential

influence of R_i in period T is the combination of S_i and H_i . The combined influence of R_i on the sampling time per unit time t can be quantified as ($t \leq T$), which is the weighted sum of aby.

Therefore, we present the Impact Maximization Quantization Model (IMM) formula (1), which uses the influence maximization approach to maximize the sampling of the stream. Since the time is continuous, first let t be the unit time length, where $t \leq T$, let $l = T/t$, which means that there are 1 unit Slot in the T period, and each slot is recorded as $s^1 \dots s^l$. Where S_i represents the set of Slots allocated for R_i . When $|S|$ is larger, the comprehensive influence of the nodes is larger, but (3) gives the comprehensive influence of the nodes with Slot. The number of judgment conditions is increased. When $v_i * |S| > |F_i^c|$ is: the node realizes full capture of the current stream passing through it, then more Slot allocation will not continue to enhance its dynamic influence, and will only continue to enhance the potential impact. force. Therefore, the combined influence of all nodes is summed, and the dynamic influence of the overlapping part of the whole system is subtracted to obtain the total influence of the system.

$$\max \sum_i^n \left(\alpha \cdot \frac{\delta(v_i, |\tilde{S}_i|)}{|F^c|} + \frac{|\tilde{S}_i|}{T/t} \cdot (\beta \cdot S_i + \gamma \cdot H_i) \right) \quad (1)$$

$$- \frac{\alpha}{|F^c|} \cdot \sum_{f_k \in F^c} \sum_{l=1}^{T/t} (P\{N_p^k(t) > 0\} \cdot \psi(f_k, s^l))$$

subject to:

$$\delta(v_i, |\tilde{S}_i|) = \begin{cases} v_i \cdot |\tilde{S}_i|, & v_i \cdot |\tilde{S}_i| < |F_i^c| \\ |F_i^c|, & \text{ELSE} \end{cases} \quad (2)$$

$$U = \{R_i, f_k \in F_i^c \wedge s^l \in \tilde{S}_i\} \quad (3)$$

$$\psi(f_k, s^l) = \begin{cases} |U| - 1, & |U| \geq 1 \\ 0, & |U| = 0 \end{cases} \quad (4)$$

$$\sum_i^n f(\tilde{S}_i) \leq K, \quad f(\tilde{S}_i) = \begin{cases} 1, & |\tilde{S}_i| \geq 1 \\ 0, & |\tilde{S}_i| = 0 \end{cases} \quad (5)$$

$$\sum_i^n w_i \cdot \varphi(\tilde{S}_i, s^l) \leq C \cdot t, \forall s^l \wedge \varphi(\tilde{S}_i, s^l) = \begin{cases} 0, & s^l \notin \tilde{S}_i \\ t, & s^l \in \tilde{S}_i \end{cases} \quad (6)$$

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

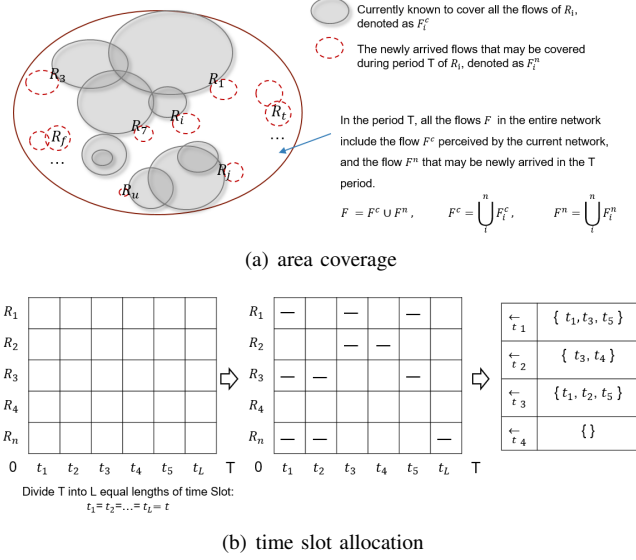


Figure 2: overview of model

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 \tilde{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.

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 D_i indicates the influence of the node based on the intermediate degree of the flow. D_{ik} indicates the dynamic influence of node i when the k th node is selected during the K th round selection. In the K th 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} .

$$c_{max} = \max\{c_j \mid c_j = \sum_i m_{ij}, i \in [1, I], j \in [1, J] \wedge i, j \in Z\} \quad (7)$$

$$D_i^k = \left| F_i - \bigcup_c^{k-1} \widetilde{F}_c \right| / \left| F - \bigcup_c^{k-1} \widetilde{F}_c \right| \quad (8)$$

$$H_i = TF_i / TF \quad (9)$$

$$S_i = C_i / \sum_i^n C_i \quad (10)$$

$$I_i^k = \alpha \cdot D_i^k + \beta \cdot S_i + \gamma \cdot H_i \quad (11)$$

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

```

1: define  $R^s = \{\}$  // The Set of Selected Routers
2: for  $k = 1; k < K; k++$  do
3:   for each  $R_i \in R - R^s$  do
4:     if  $I_i^k > max$  then
5:        $max = I_i^k$ 
6:        $SR = R_i$ 
7:     end if
8:   end for
9:   put  $SR$  to  $R^s$ 
10:  mark  $SR$  as  $R_k$  in  $R^s$ 
11: end for
12: return  $R^s$ 

```

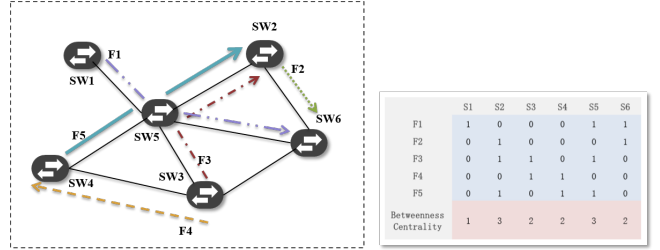


Figure 3: Intermediary center based on the number of streams

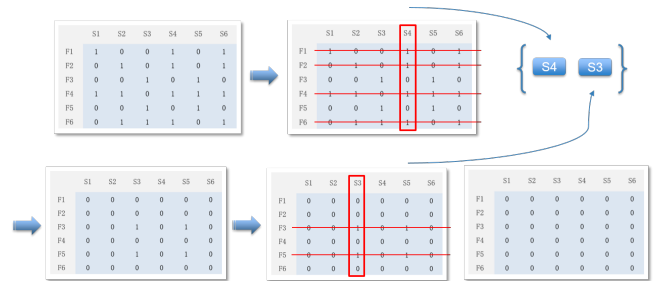


Figure 4: Sampling Point Selection

3.3. Allocation of Time Slot

After the nodes are selected the number of slots needs to be allocated to these nodes. For R_i , the number of selected slots is not linearly related to the value. Equation (3) describes the relationship between them. The value function $\alpha \cdot \frac{\delta(v_i, |\widetilde{S}_i|)}{|F_c|} + \frac{|\widetilde{S}_i|}{T/t} \cdot (\beta \cdot S_i + \gamma \cdot H_i)$ increases as the number of Slots increases in $Mode(1)$. When the ELSE in

equation (3) is satisfied, there will be another growth trend, which depends on its S_i and H_i value. The reason is that if a node captures 100% of the flows it currently covers, more Slot allocations will not generate more value for the current flows capture, but only increase the potential flows capture. Therefore, this cannot use the multi-catch algorithm to solve the optimal solution. The problem can be solved separately in two parts, the first part is found in formula (3), when each node satisfies ELSE. In this case, the number of optional Slots for each node can be determined, and then the problem is converted into a multi-clip problem to solve the optimal solution for a total of K items to be selected (Rs1-Rsk), each type of candidate to be selected There are N_i^l , and the return of each item in one unit Slot is $v(R_i, t)$, the cost is $w(R_i, t)$, and the constraint is C. After the first part is solved, the remaining C, as the constraint of the second part, is still a multi-clip problem. At this time, the number of optional slots of each node is $\frac{T}{t}$ minus the first part. The number of nodes that have been assigned. Each part can get the optimal solution of each part. Solving the complexity of the problem $O(\frac{T}{t} \cdot |R^s| \cdot C)$ When T is too large or t is too small and C is too large, the time to solve is unacceptable.

We give a simpler and more efficient to achieve, combines the two processes into the same process, using simple influence from high to low polling distribution, so that The current high-impact node of preferentially allocates a Slot. When the node satisfies the ELSE part of the formula (3), the influence prioritization in the subsequent allocation process only uses $\frac{1}{T/t} \cdot (\beta \cdot S_i + \gamma \cdot H_i)$, and the nodes that does not satisfy the formula (3) ELSE, Continue to use the comprehensive influence to sort. This algorithm is consistent with the formula definition in Model(1).

For the current influence of a node under one unit slot t , a Slot is assigned from high to low, and each round is allocated, and the influence of each node is reordered (because some nodes will satisfy the ELSE condition of formula (3), and Causes a change in influence) until $C = 0$ or cannot be assigned.

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, 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 $|\tilde{S}_i \cap \tilde{S}_j|$ represents the number of overlaps of the time slots of the two nodes, and $|\tilde{F}_i \cap \tilde{F}_j|$ 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.

Algorithm 2 Impact Priority Polling Allocation Slots

```

1: Define  $CNT[1..K] = 0$ ;  $\tilde{I}[1..K] = 0$ ;
2: while  $C > 0$  OR  $C$  is different from the last round do
3:   for each  $R_i^s \in R^s$  do
4:     if  $R_i$  satisfy  $\delta(v_i, |CNT[i]|)$  the ELSE condition then
5:        $\tilde{I}[i] = \frac{1}{T/t} \cdot (\beta \cdot S_i + \gamma \cdot H_i)$ 
6:     else
7:        $\tilde{I}[i] = \alpha \cdot \frac{v_i \cdot 1}{|F^c|} + \frac{1}{T/t} \cdot (\beta \cdot S_i + \gamma \cdot H_i)$ 
8:     end if
9:   end for
10:  Descending sorting  $\tilde{I}$ ;
11:  for  $i = 1$ ;  $i < K$ ;  $i++$  do
12:    if  $C \geq w_i$  then
13:       $CNT[i]++$ 
14:       $C = C - w_i$ 
15:    end if
16:  end for
17: end while
18: return  $CNT$ 

```

$$\min \sum (|\tilde{S}_i \cap \tilde{S}_j| \cdot |\tilde{F}_i \cap \tilde{F}_j|), \forall i, j \wedge i \neq j \quad (12)$$

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 gives the steps. In the previous section, the CNT array has been calculated. At the beginning of the algorithm, the M^{slot} two-dimensional array is initialized to store the placement relationship between R_i and s^l : $M^{slot}[i][l] = 1$, which represents the R_i node sampling at s^l . In each round, a slot is selected for the node whose CNT is not 0, and the total number of coverages of the stream generated after the selected slot is selected is the minimum value of all the optional slots. Therefore, through each round, for each node whose CNT is not 0, a Slot that minimizes the total number of coverages of the current entire system stream is arbitrarily placed until all the Slots of the nodes are placed (all CNT are 0).), we get an approximate optimal solution. In the solution process, the $H[l]$ array is used to store the total number of coverages of each Slot stream, and when s^l is selected once, $H[l]$ is updated. After the node R_i selects s^l , the new flow cover total of s^l is calculated as: $H[l] = \sum_{j=1 \wedge j \neq i}^K (|\tilde{F}_i \cap \tilde{F}_j| \cdot M^{slot}[j][l]) + H[l]$. That is, in equation (17), $|\tilde{S}_i \cap \tilde{S}_j| = 1$, and j is the node that has been selected for placement in s^l . The demonstration of the whole algorithm is shown in Fig. 6. In the example, the approximate solution we solved by this algorithm is 35, and the optimal solution is 33.

Algorithm 3 Order of Time Slot Based on Greedy

Input: M, S, c_j

```

1: while  $CNT[i] > 0, \exists i \wedge i = 1, 2, \dots, k$  do
2:   for  $i = 1; i \leq K; i++$  do
3:     if  $CNT[i] > 0$  then
4:        $Min = Max\ Integer$ 
5:       for  $l = 1; l < \frac{T}{t}; l++$  do
6:         if  $M^{slot}[i][l] = 0$  then
7:            $temp = \sum_{j=1 \wedge j \neq i}^K (|\tilde{F}_i \cap \tilde{F}_j| \cdot M^{slot}[j][l] + H[l])$ 
8:           if  $temp < Min$  then
9:              $Min = temp; Sp = l$ 
10:          end if
11:        end if
12:      end for
13:       $M^{slot}[i][Sp] = 1; CNT[i]--; H[Sp] = Min$ 
14:    end if
15:  end for
16: end while
17: return  $M^{slot}$ 

```

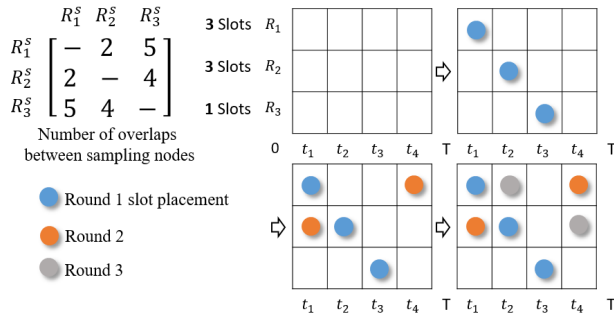
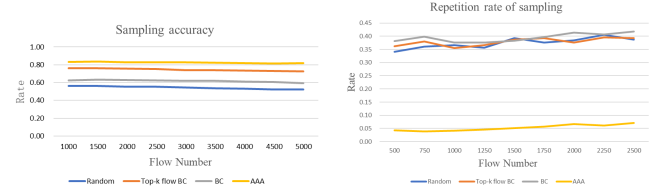


Figure 5: Illustrating sampling node slots placement based on greedy algorithm

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 openswitch + 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

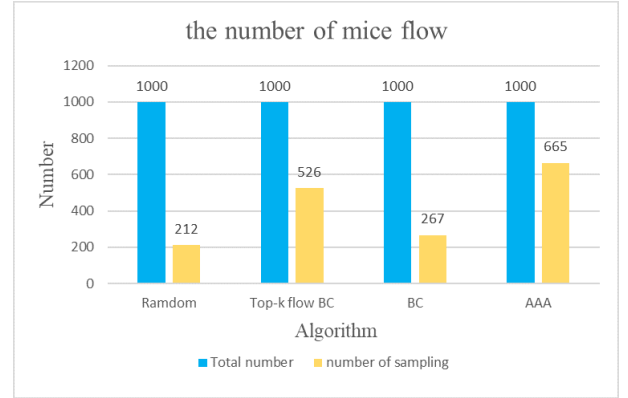


Figure 7: comparison of different algorithms in number of mice flow

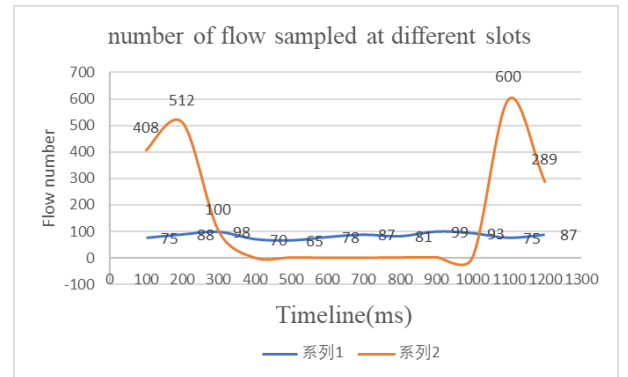


Figure 8: comparison of sampling flow number at different slot

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. Christodouloupoulos, G. Landi et al, NEPHELE: An End-to-End Scalable and Dynamically Reconfigurable Optical Architecture for Application-Aware SDN Cloud Data Centers, in *IEEE Communications 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.