AAA: High Agile Adaptive Application-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.

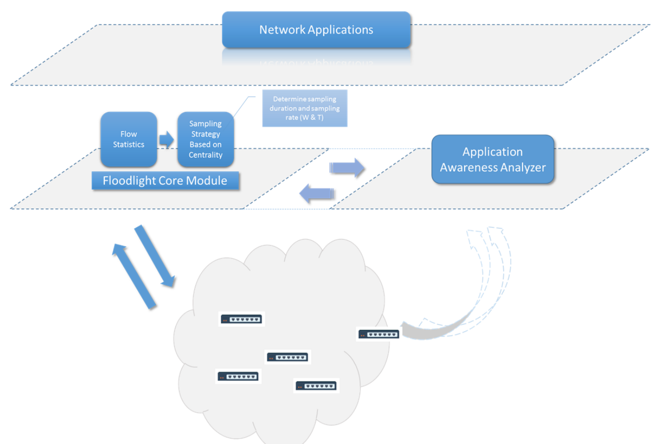


Figure 1: The System Architecture Of AAA

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

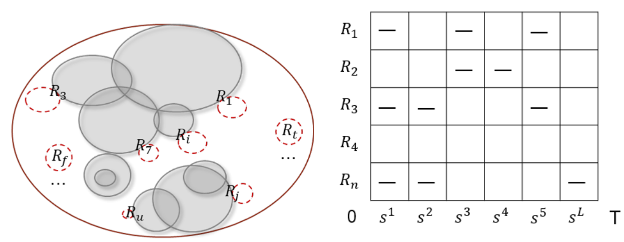
3.1. Problem of Model

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 Ri, 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.



(a) area coverage (b) time slot allocation

Figure 2: overview of model

3.2. Sampling Point Selection

In large-scale DC networks, while all the switches sample flows, the excessive frequency of sending flow tables and group tables will cause extra overhead for SDN controller. Therefore, we need a strategy that can select a small number of most influential sampling nodes so that the number of flows sampled by these nodes is close to or equal to the global nodes. We may find the situationis is very similar to the graph theory. In a subnet topology, based on the concept of graph theory, we regard switches as nodes and regard flow paths as edges. Therefore, we define a flow information matrix. Then we calculate betweenness centrality and the most influential nodes has the maximal betweenness centrality. As show in Algorithm 1, the strategy is called sampling node election algorithm based on betweenness centrality.

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}.

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

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 $R_i \in R - R^s$ **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
  $F_k^s = F_i - \bigcup_{c=1}^{k-1} \widetilde{F}_c$
end for
return R^s

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

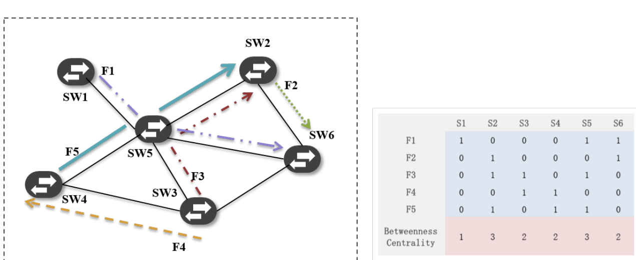


Figure 3: Intermediary center based on the number of streams

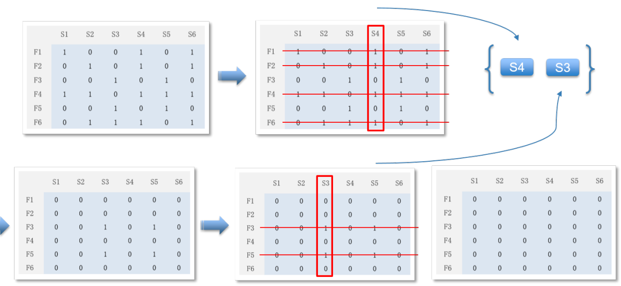


Figure 4: Sampling Point Selection

Algorithm 2 Sampling Point Selection Based on Centrality Measure

Input: M , S , c_j
while $M! = O$ **do**
  if c_{max} = c_j **then**
    Selecting sw_i
  end if
  Puting sw_j into S;
  for all f_i which $m_{ij} = 1$ **do**
    for all sw_j which $j \in [i, J]$ **do**
      if $m_{ij} = 1$ **then**
        $m_{ij} = 0$
      end if
    end for
  end for
end while
return S

3.3. Allocation of Time Slot

Algorithm 3 Allocation of Time Slot Based on XXX

Input: M , S , c_j
return S

3.4. Order of Time Slot

Algorithm 4 Order of Time Slot Based on XXXXX

Input: M , S , c_j
return S

4. Experiments and results

- Schematic diagram of strategy
- Algorithm
- Union/Find Grouping

5. Conclusion

- Lab environment

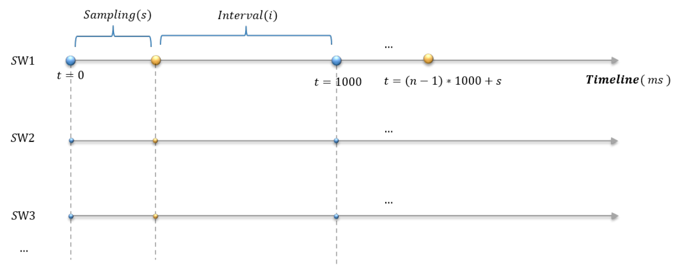
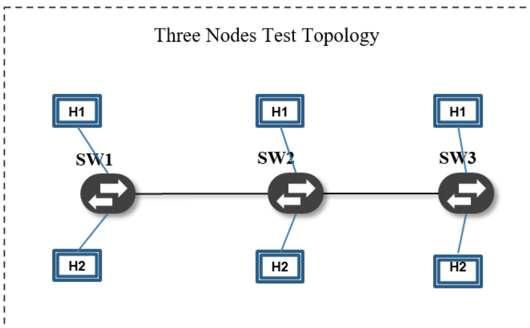


Figure 5: Simultaneous Sampling



(a) Test Toplogy

Flow Information Matrix (M)

	S1	S2	S3
F1	1	1	1
F2	1	1	1
F3	1	1	0
F4	1	1	0
F5	0	1	1
F6	0	1	1
F7	1	0	0
F8	1	0	0
F9	0	0	1
F10	0	1	0

(b)Flow Information Matrix

Figure 6: This is total name.

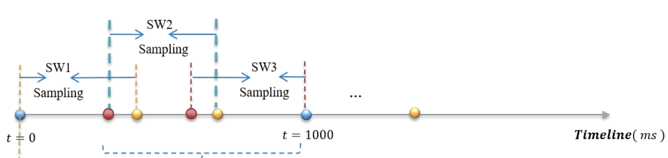


Figure 7: Even time-division sampling

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

TABLE 2: 110 Switches & 22 Hosts & 1400 Flows Comparison In Real Topology

Strategy	Captured	Sampling Accuracy	Repeat Rate
Single Time Sampling	1069	0.764	36.05%
Even-Divsion Time Sampling	1094	0.7814	12.86%
One More Back-up Sw	1152	0.7979	15.78%

TABLE 3: Cost Matrix

	S1	S2	S3
S1		4	2
S2			4
S3			

algorithm

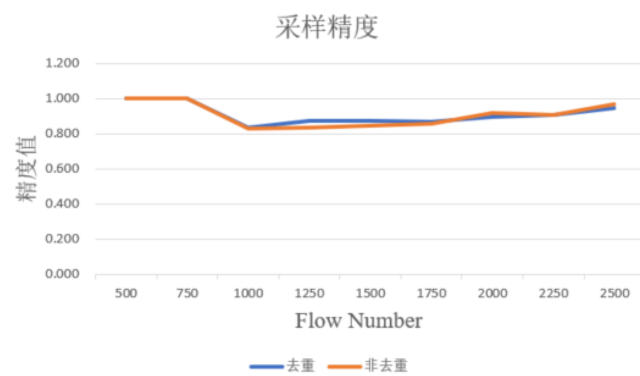


Figure 8: The definition of agile application-aware network

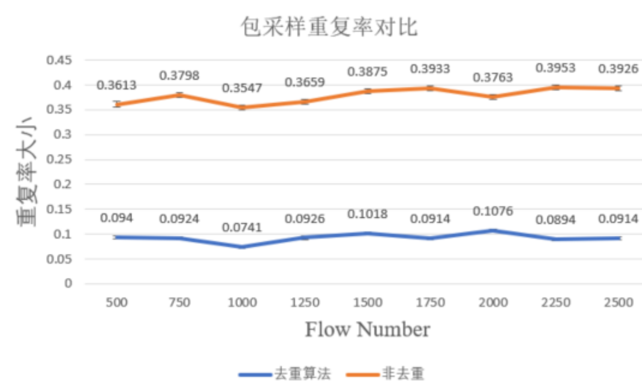


Figure 9: The definition of agile application-aware network

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.

TABLE 4: Comparison

Sampling Order	Cost	Accuracy	Packet Repeat Rate
(S1,S2,S3)	8	100.0%	20.46%
(S3,S1,S2)	6	100.0%	15.50%

TABLE 5: 15 Switches & 30 Hosts & Around 1000 Flows Comparison In Test Bed

Strategy	Packet Repeat Rate
Greedy Algorithm	4.39%
Random	5.93%

[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.

TABLE 6: 110 Switches & 22 Hosts & Around 1000 Flows Comparison In Real Topology

Strategy	Packet Repeat Rate
Greedy Algorithm	6.03%
The Most Bad	7.54%
Random	7.11%