Deploying DeepRoute on Chameleon (Demo Description)

Bashir Mohammed
Scientific Data Management Group(SDM)
Lawrence Berkeley National Lab
Berkeley,CA, USA
bmohammed@lbl.gov

Mariam Kiran
Energy Sciences Network(ESnet)
Lawrence Berkeley National Lab
Berkeley,CA, USA
mkiran@es.net

***Abstract*—Current WAN traffic presents difficult challenges to manage traffic surges and bandwidth management on network infrastructure. Some existing traditional centralized network path calculation approaches are often not scalable, where distributed approaches are not fully aware of the end-to-end performance. The recent advances in deep reinforcement learning approaches have shown promise in solving complex dynamic network routing path optimization problems.**

In this demo, we develop and explore a deep reinforcement learning approach applied to a real network path optimization problem with high dimensional state and action space. Specifically, we create an environment on Chameleon Cloud network testbed and deploy reinforcement learning algorithms - namely Random, Q-Learning and Deep Q-Network (DQN) respectively. The goal is to predict the most optimal network path between two distributed nodes across two locations. Without having any prior information or knowledge of the environment, the adaptive agent learns optimal paths in real-time between two nodes from source to destination.

Resources and Tools Required

Note: This experiment uses the Bring-Your-Own-Controller (BYOC) feature provided by Chameleon. For more details on how to run SDN experiments on Chameleon go to[1].

A. Bare Metal Nodes

- One Client(Compute Node-CHI@UC)
- One Server (Compute Node-CHI@TACC)
- Two Cross Traffic
- One controller(CHI@TACC)
- All Compute Nodes - CC-Ubuntu 16.04
- Controller Node - CC-CentOS7

B. Network

- AL2S by Internet2
- Two Corsa switches: TACC and UC
- Direct Stitch Link via Exogeni. For more details on how to stitch go to[2].

C. CNERT's Workshop Default Demo Set-up is sufficient.

Figure 1, shows the topology of our environment and Table I shows the settings of our environment. For more details and steps on how to setup the environment and perform experiment, go to[3].

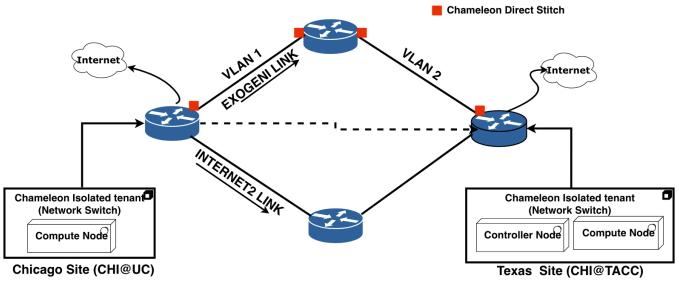

Fig. 1: Network Topology

TABLE I: Chameleon Core Network Environment settings.

Node Specification	Node Location on Chameleon		
	CHI@UC	CHI@TACC-1	CHI@TACC-2
IP Address	192.168.100.17	192.168.100.29	10.52.3.225
Floating IP	192.5.87.229	129.114.109.224	129.114.109.70
Gateway	192.168.100.1	192.168.100.1	192.168.100.1
(Direct-stitch)	VLAN1 ID:3294	VLAN2 ID:3501	–
Switches	Corsa Switch	Corsa Switch	Corsa Switch
Bare-metal	Compute Node	Compute Node	Controller Node
OS	CC-Ubuntu16.04	CC-Ubuntu16.04	CC-CentOS7

Experiment Procedure

We built a customized gym called deeproute-gym which is our simulated environment where our reinforcement learning agent tries different actions and gets a reward. In order to further test the deeproute-gym we emulate a WAN network(from source to destination) with two alternate paths using Chameleon cloud network test-bed for our experiments as shown in Figure 1. Our topology has multiple network paths across two different distributed sites at the University of Chicago and Texas Advanced Computing Center (CHI@UC to CHI@TACC), via Exogeni test-bed and pointing towards Internet2's advanced layer 2 service (AL2S) socket. We define three metrics for our observation space - bandwidth, latency and flow completion time. We created two VLANs: one

[1]https://tinyurl.com/ta79hd9/
[2]https://tinyurl.com/u7yk47p/
[3]https://github.com/esnet/daphne-public/

on each chameleon site and a stitch-port via Exogeni. The capacities (bandwidth) are setup as 1 x 10 Gbps and 1 X 8 Gbps for network path1 and path2. We initiate a ping request from CHI@UC to CHI@TACC using ICMP protocol and record the time in which the ping reaches the destination. We calculated a total of 15-hops across the internet2 AL2S link and 1-hop across the direct-stitch link via exogeni. The controller selects paths by learning which egress port to use. To have a global view, we configure a Ryu network controller at the CHI@TACC node, Openflow switches using Corsa Switches at both CHI@UC and CHI@TACC. All instances are configured on a bare metal node running on Ubuntu Linux host while the Controller is configured on CentOS, all running on real network. For dynamic multipath selection, we send a packet via one of the network paths. The flows go through paths: CHI@UC to Exogeni to CHI@TACC using VLAN1 or CHI@UC to Internet2 to CHI@TACC via VLAN2. We use link layer discovery protocol to obtain link and switch states in the topology to advertise device identity and abilities. The procedure to reproduce this experiment is as follows:

- Obtain resources on Chameleon and perform the setup as explained on the **README.md** file here[4].
- Login to your Controller node at TACC and Clone this repo[5].
- To run the experiments Install all the dependencies and from your terminal run: deeproute_rl_dqn_agent.py and deeproute_rl_random_agent.py separately.

[4]https://github.com/esnet/daphne-public/blob/master/Chameleon/README.md/
[5]https://github.com/esnet/daphne-public/