APPENDIX A ARTIFACT DESCRIPTION

We present the reproducibility artifact of our experimental validation. The source code is available in our GitHub repository¹.

A. Overview

- Matching game framework. See Section A-B;
- Dataset. See Section A-C;
- Software dependencies. See Section A-D;
- Run-time environment: Visual Studio Code IDE 1.65.2;
- Installation. See Section A-E;
- Testbed. See Section A-F.

B. Matching game framework

Matching game is a Python-based framework² for solving matching game based models. We used YAML files containing device and microservice preference lists (DPL.yaml and MPL.yaml), and the schedules of microservices on devices (matching-testbed.yaml).

C. Data sets

Two datasets for training neural network and machine learning models:

- Case study I: German traffic sign benchmark³;
- Case study II: Amazon reviews for sentiment analysis⁴.

D. Software dependencies

1) Model:

- Python libraries networkx, operator, numpy, yaml, json.
- 2) Integrating C^3 -MATCH to Kubernetes scheduler:
- Docker v20.10.12⁵ on all devices and instances;
- Orchestrating by Kubernetes v1.21;
- Measuring bandwidth and latency between the local Edge devices: kube-latency⁶;
- Monitoring local Edge cluster by the Prometheus operator v0.45.0⁷;
- ullet Python script "scheduling.py" collects the monitoring information by Prometheus Python API⁸ from the local Kubernetes Edge cluster, and then utilizes the Python client library v17.17 for Kubernetes⁹ to execute the customized C^3 -MATCH scheduler and deploy the application pods on the appropriate devices;

- Transmitting data through asynchronous message queue platforms: 1) a Kubernetes-based KubeMQ v2.2.10¹⁰ for the local Edge cluster, and 2) ZeroMQ v22.3.0¹¹, to receive data by the Cloud and Fog instances.
- Ping and iPerf3 tools to benchmark the latency, maximum achievable bandwidth, and effective downlink throughput between instances.
- 3) Case studies: We Dockerized all the microservices with Docker $v19-20^{12}$.
- *a) Python libraries:* ffmpeg, scikit-learn, numpy, tensorflow, keras, matplotlib, opency-python, pandas.
- b) Tensorflow API: Docker image nvcr.io/nvidia/14t-tensorflow r32.5.0-tf2.3-py3¹³ on Nvidia devices.

E. Installation

- Matching library by package installer for Python 3.9 (pip3.9);
- Docker Engine on Ubuntu¹⁴, Kubernetes on Ubuntu¹⁵, and Kubernetes on (vanilla) Raspbian Lite¹⁶;
- NVIDIA Jetson Linux¹⁷ and Raspberry Pi OS¹⁸.

F. Testbed

- Three on-demand instances from the Exoscale provider¹⁹, hosted in the data center of the A1 network operator in Sofia, Frankfurt, and Vienna: large with four virtual cores and 8 GB of memory; medium with two virtual cores and 4 GB of memory; small with two virtual cores and 2 GB of memory.
- A private Cloud at the university campus and medium Edge instances, managed by an OpenStack v13.0 and Ceph v12.2 with support for block and S3compatible object storage;
- Five NVIDIA Jetson Nano (NJN) running Linux for Tegra (L4T) OS, 10 Raspberry Pi-3 model B+ (RPi3) and 30 RPi4 running Raspberry Pi OS.
- Testbed components are interconnected by a managed layer-3 HP Aruba with $48~1~\mathrm{Gbit\,s^{-1}}$ ports, $3.8~\mu\mathrm{s}$ latency and aggregate throughput of $104~\mathrm{Gbit\,s^{-1}}$.

¹https://github.com/SiNa88/C3-Match/

²https://github.com/daffidwilde/matching

³https://www.kaggle.com/meowmeowmeowmeow/gtsrb-german-tra ffic-sign

⁴https://www.kaggle.com/bittlingmayer/amazonreviews

⁵https://www.docker.com/

⁶https://github.com/simonswine/kube-latency

⁷https://github.com/prometheus-operator/prometheus-operator

⁸https://pypi.org/project/prometheus-api-client/

⁹https://github.com/kubernetes-client/python

¹⁰https://github.com/kubemq-io/kubemq-community/releases/tag/v2.2.10

¹¹https://pypi.org/project/pyzmq/

¹²https://www.docker.com/

¹³https://catalog.ngc.nvidia.com/orgs/nvidia/containers/l4t-tensorflow

¹⁴https://docs.docker.com/engine/install/ubuntu/

¹⁵https://phoenixnap.com/kb/install-kubernetes-on-ubuntu

¹⁶https://github.com/alexellis/k8s-on-raspbian/blob/master/GUIDE.md

¹⁷https://developer.nvidia.com/embedded/linux-tegra

¹⁸https://www.raspberrypi.com/software/

¹⁹ https://www.exoscale.com/compute/