Supplementary File

Deliang Yi, Xin Zhou, Yonggang Wen, Rui Tan

◆

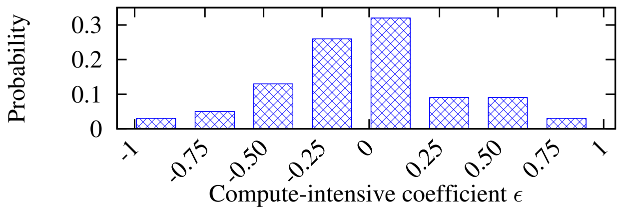


Fig. 1. Distribution of the compute-intensive coefficient ϵ. An ϵ value is computed for each job; a smaller absolute value of ϵ means that the job better conforms to the compute-intensive assumption. The y-axis is the probability that an ϵ falls within a bin on the x-axis.

This document includes the supplementary materials for the paper titled "Efficient Compute-Intensive Job Allocation in Data Centers via Deep Reinforcement Learning."

APPENDIX A
NSCC DATASET PROFILES

As discussed in §4.1 of the paper, under the compute-intensive assumption, we have $u_i(k+1) = u_i(k) + \frac{m_i(k)}{n_i}$. Thus, we measure a coefficient $\epsilon = u_i(k+1) - u_i(k) - \frac{m_i(k)}{n_i}$ for each allocated job to assess the validness of this assumption. If the assumption holds perfectly, we should observe $\epsilon = 0$. Fig. 1 shows the distribution of all jobs' ϵ values, which exhibits a Gaussian-like shape. This means that a majority of jobs achieve high utilization of the assigned cores. However, the core utilization behaviors of the computing jobs can be complex. For instance, a newly admitted job may contend for low-speed resources (e.g., hard disks) with existing jobs, leading to a reduction of the overall processor utilization. The deviations from the ideal compute-intensive assumption shall be addressed by the first LSTM network.

Fig. 2 shows the processor temperature and the server power consumption traces when a job is allocated to the server. We can see that both the temperature and power consumption increase during the job execution period between the two vertical dash lines. After the completion of the job, the temperature and power consumption drop gradually. This shows the temporal dynamics of the processor temperature and power consumption processes. Such dynamics shall be captured by the second LSTM network.

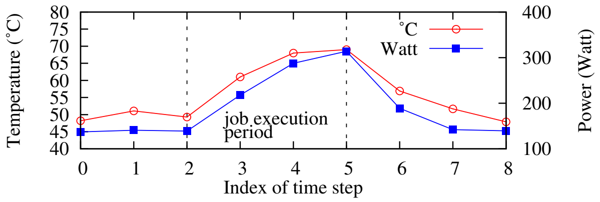
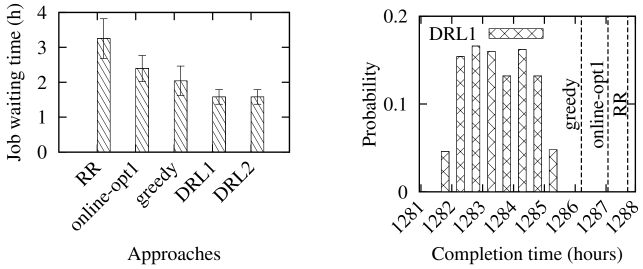


Fig. 2. Impact of a job on processor state. The x-axis is the index of time step; the left y-axis is the processor temperature; the right y-axis is the processor power consumption.



(a) Waiting time
(b) Completion time

Fig. 3. Job waiting time & completion time. (a) The x-axis gives various approaches; the y-axis represents the waiting time before a job is allocated; the error bar represents s.d. (b) The x-axis represents job completion time; the y-axis represents the probability distribution under *DRL1*; the vertical dashed lines represent the job completion times under other approaches.

APPENDIX B
JOB PROCESSING THROUGHPUT

From our investigation on the distribution of the time interval between the arrivals of any two consecutive jobs over the period of 52 days, only 3% of the intervals are smaller than 10 minutes. Each job allocation approach operates on a first-come, first-served (FCFS) basis. If a job at the front of the job queue cannot find an eligible processor with sufficient spare cores, the job allocator will wait until an eligible processor becomes available. Thus, the jobs' waiting times from arrival to allocation and the completion time of all the 1,500 jobs characterize the job processing throughput under a certain job allocation approach. Note that *DRL1* and *DRL2* allocate a single job and two jobs each time, respectively. Other baseline approaches allocate a single job each time. To account for the DRL's randomness, we conduct the simulation of allocating the 1,500 jobs for 500 times.

Fig. 3(a) shows the waiting time for each job under different allocation approaches. *DRL1* and *DRL2* achieve

---

- *D. Yi, X. Zhou, Y. Wen, R. Tan are with School of Computer Science and Engineering, Nanyang Technological University, Singapore.*
E-mail: {yideliang, zhouxin, ygwen, tanrui}@ntu.edu.sg

similar job waiting times because both them continuously allocate jobs until the queue is empty. With our DRL-based job allocators, the jobs can be promptly allocated to the processors for execution. With RR, the jobs experience the longest waiting time. We also measure the completion time of all jobs. Fig. 3(b) shows the distribution of the completion time with *DRL1*. The completion time is up to 1,286 hours. The vertical lines in the figure represent the completion times with other baseline approaches. We can see that *DRL1* has slightly shorter completion time (i.e., higher job processing throughput) compared with other baselines. Therefore, our DRL-based job allocator achieves slightly higher job processing throughput.

Note that if some other job in the queue that requests less cores can be allocated before the job at the queue front that cannot be allocated due to lack of eligible processors, the job waiting times and the completion times can be reduced. Our future work will study how to extend the DRL formulation to address such non-FCFS schemes. Moreover, the extensions to address specified job priority and job soft deadlines are also interesting topics for future research.