Cloud Resource Allocation and Power Management using Deep Reinforcement Learning

May, 2019

Hongyi Guo, Shenglong Ye

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Contents

1 Background

2 Introduction

3 Model

4 Experiments

5 Future Work

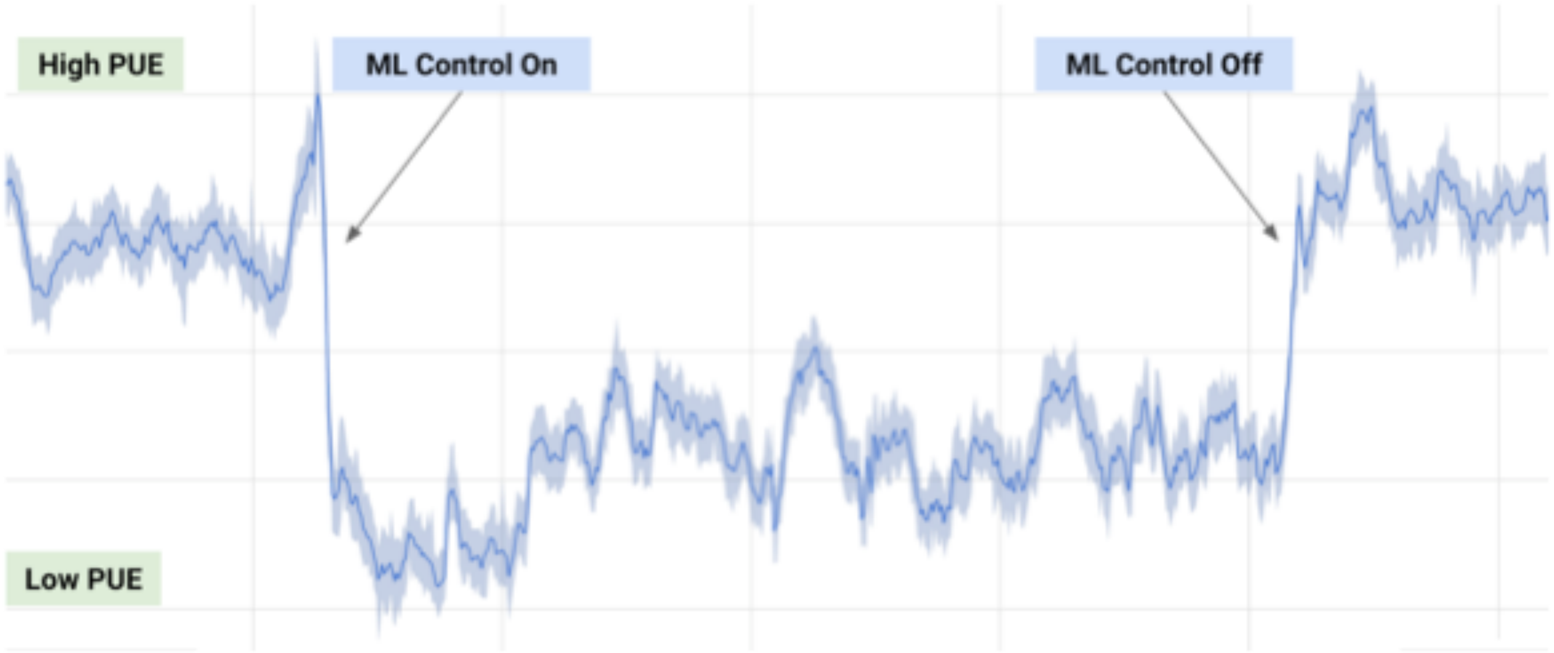
上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Related work

- Machine Learning Applications for Data Center Optimization (DeepMind, 2016)
  - Using a system of neural networks trained on different operating scenarios and parameters within our data centres, they created a more efficient and adaptive framework to understand data centre dynamics and optimize efficiency.
  - We accomplished this by **taking the historical data** that had already been collected by thousands of sensors within the data centre -- data such as temperatures, power, pump speeds, setpoints, etc. -- and using it to train an ensemble of deep neural networks.

Related Work

- Machine Learning Applications for Data Center Optimization (DeepMind 2016)



Introduction

- We focus on both
  - saving the power energy of the cloud server
  - reducing the job latency
- Reference
  -

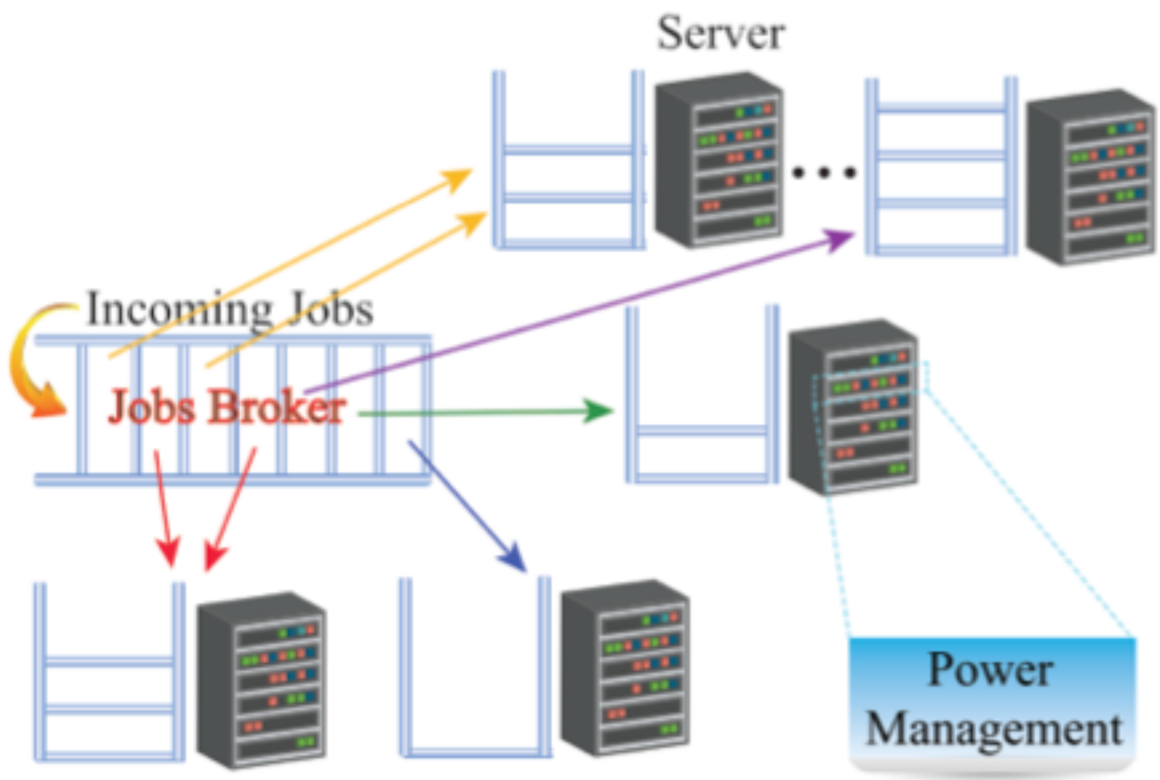
A Hierarchical Framework of Cloud Resource Allocation and Power Management Using Deep Reinforcement Learning

Ning Liu*, Zhe Li*, Jielong Xu*, Zhiyuan Xu, Sheng Lin, Qinru Qiu, Jian Tang, Yanzhi Wang
Department of Electrical Engineering and Computer Science,
Syracuse University, Syracuse, NY 13244, USA
{nliu03, zli89, zxu105, jxu21, shlin, qiqiu, jtang02, ywang393}@syr.edu

Model

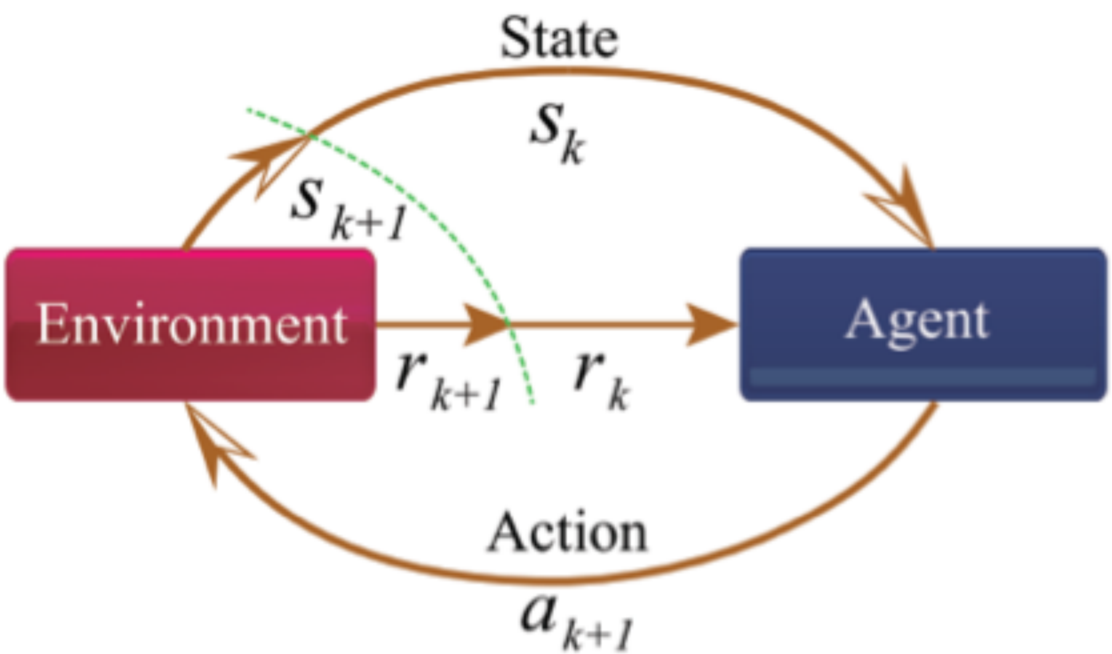
- A **Hierarchical** Framework of Cloud Resource Allocation and Power Management Using **Deep Reinforcement Learning**
  - The proposed hierarchical framework comprises:
    - ***a global tier*** for **job allocation** to the servers
    - ***a local tier*** for **distributed power management** of local servers.
- According to paper, the proposed framework can achieve the best **trade-off between latency and power/energy consumption** in a server cluster.

Model



Model

- Deep Reinforcement Learning



Local Tier

- Local Tier is responsible for controlling the turning ON/OFF of local servers in order to simultaneously
  - Reduce the power consumption
  - Reduce the average job latency
- The local tier includes
  - the ***workload predictor*** using **LSTM**
  - the adaptive ***power manager*** based on the model-free, continuous-time **Q-learning / Actor Critic.**

Power manager

- Deep RL based power manager
- *Decision epoch*
  - Whenever the machine enters the idle state (usage = 0) and there is no waiting job in the pending queue, power manager should decide how long to shut down this server for.
- *Action*
  - How long we shut down the server for, or zero (keep the server on).

Power manager

- Deep RL based power manager
- ***State***
  - Power consumption state
  - The estimated next job inter-arrival time from the work-load predictor
- ***Reward:***
  - $r(t) = -w * P(t) - (1 - w) * JQ(t)$
  - $P(t)$: power consumption
  - $JQ(t)$: number of jobs buffered in the pending queue
  - w: the weighting factor

Global Tier

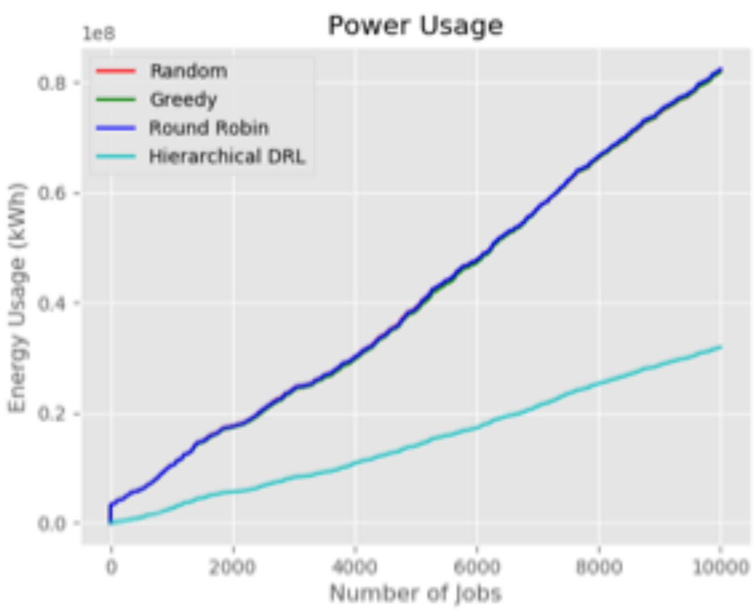
- Decision epoch: Every time a new job arrives
- Use greedy method to allocate jobs to servers.
  - Allocate current job to the most idlest server.
  - Since CPU is the largest constrain for these servers, we define the idlest server to be the server with the lowest CPU usage percentage.
- The referenced paper used a DQN to select the server that our task is to be allocated to. But we thought it brings more troubles than benefits. And we will explain why next.

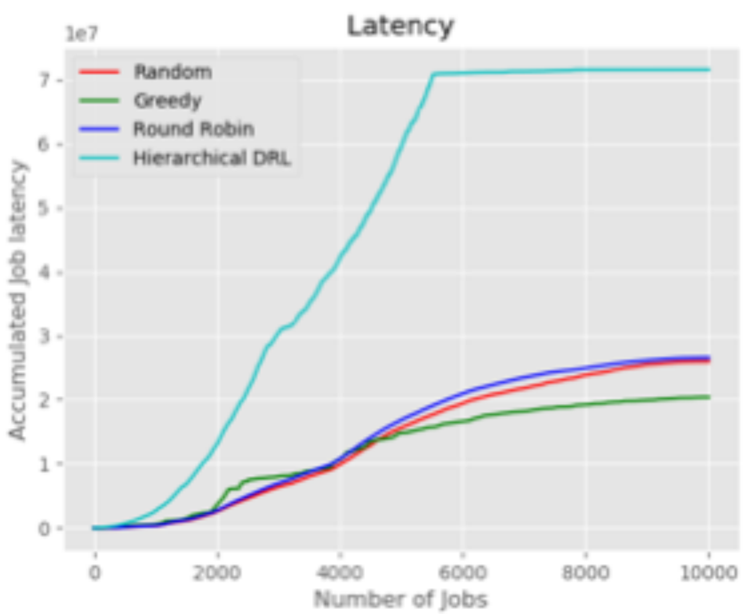
Deficiency

- Too many deep learning models in this architecture
  - Rather difficult to train
  - Interact
- If we dig a little bit deeper ...
  - Non-stationary environment
  - ...
- Multiple tasks coming at the same time are very possible to be dispatched to the same server.

Experiments last week

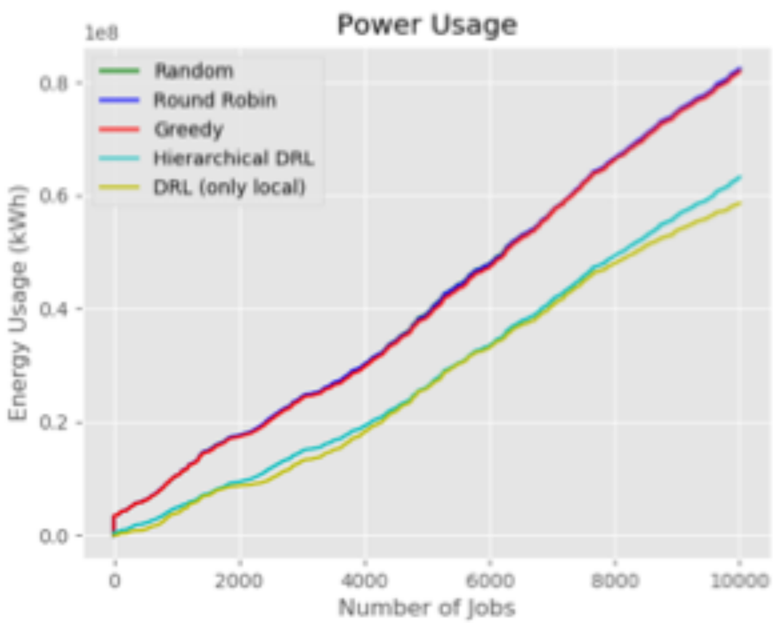
- We have implemented the framework introduced by the paper. Here we illustrate the **power usage** and the **job latency** compared with three baselines as following:
- Dataset: **alibaba_clusterdata_v2018**

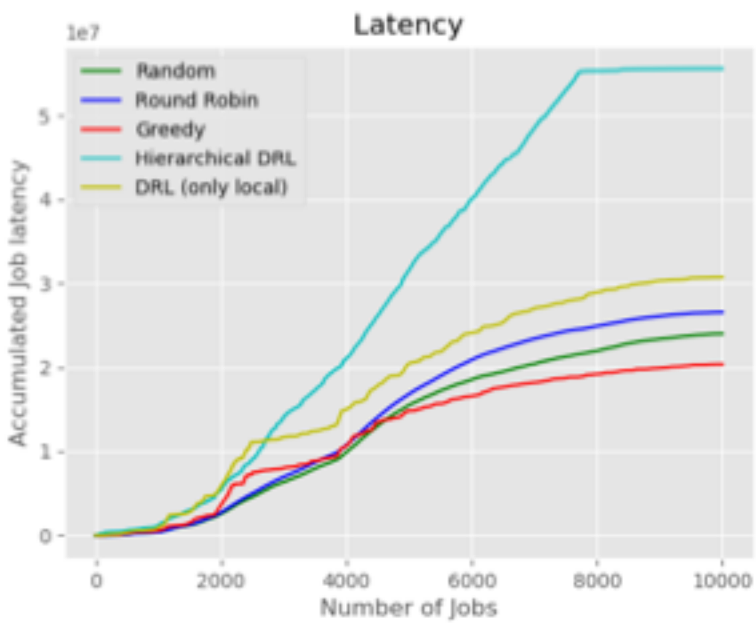




Experiments this week

- We have implemented the framework introduced by the paper. Here we illustrate the **power usage** and the **job latency** compared with three baselines as following:
- Dataset: **Alibaba_clusterdata_v2018**





Future work

- Try different methods to enhance the global tier, not just rule-based ones.
  - Bi-LSTM
  - …
- Use more advanced Reinforcement Learning models, such as DDPG, A2C, etc.
  - DDPG can handle continuous action space, while Q-Learning cannot

Thanks !