Efficient Provisioning of Virtual Machine Sets with Placement Constraints in IaaS Clouds with Deep Reinforcement Learning

***Abstract***–Today, cloud service providers like Amazon Web Services (AWS), Google Cloud, Microsoft Azure and more provide computing capacity on demand. A fundamental problem which arises when providing these services is that of mapping requests to physical machines in such a way as to minimise wasted space and maximise the number of requests that can be served given fixed physical infrastructure. This project explores the viability of using deep reinforcement learning to determine a solution for VM mapping problems with the goal of maximising revenue for the cloud service provider.

# INTRODUCTION

Reinforcement learning uses a trial-and-error approach whereby an agent takes actions in some environment; this agent is then either rewarded or penalised for these actions to reinforce the behaviour we want the agent to learn. Over time, a value mapping can be determined between states and actions. In complex environment where it may not be practical to map every state to the optimal action, we can introduce an artificial neural network (ANN)to estimate the value of states based on the values of similar states we have seen in the past. The addition of an ANN is what distinguishes deep reinforcement learning from reinforcement learning.

I believe that we can treat this as a multi-dimensional specialization of the knapsack problem in which our agent is given some number of servers each with a fixed storage, memory, and CPU capacity. At each timestep, our agent will then receive a request for VM capacity which it must then either allocate to a server, or refuse. The goal of this agent is to determine some policy for VM allocation which will maximise a cloud service provider’s revenue. I believe that this information could enable cloud service providers to tailor their pricing model based on their current capacity to encourage consumer behaviour in line with the optimal policy and this increase in efficiency of allocation could in turn allow providers to offer more competitive rates.

# SCOPE

In this paper, we are going to focus specifically on the provision of Virtual Machines (VMs), and all real-world data will be taken from Amazon’s Elastic Compute Cloud (EC2). This choice was made to both simplify comparisons and keep things consistent, and to limit the research scope to manageable levels. It is expected that other market leading cloud service providers would operate similar pricing models to AWS to remain competitive and so the learnings from this project should be transferrable.

# RELATED WORK

Previous research has looked at solving this problem through approaches like integer linear programming and heuristic-based approaches like first-fit descending

# EVALUATING PERFORMANCE

It is challenging to benchmark the success of our approach without having some data to compare it against. As detailed revenue information from EC2 is not publicly available, I propose determining some upper bound on the possible revenue generated for a given infrastructure and scoring our attempts against this. For example, we may see that a heuristic-based approach achieves 80% of the upper bound; this can give us a quantifiable target.

# MODELLING THE PROBLEM

There are two main hurdles we must overcome to model this problem. The first is finding data which is representative of the real world, and the second is determining some machine learning model which can produce meaningful insights.

I pulled real world pricing data from Amazon’s EC2 service; this was taken both from the on-demand pricing[[1]](#footnote-1) and the spot pricing[[2]](#footnote-2). In total, I collected data on the price, number of vCPUs, memory, storage, and network performance of 392instance types (instance types were ignored where pricing information for both on-demand and spot instances was not available). While we cannot know the true distribution of these instances in the real word, we can make certain assumptions. If we compare the pricing for both instance types, we can determine some discount factor which we assume to be inversely proportional to that instance type’s popularity (David Naori, 2020).

1. This is the normal pricing offered for all instances. (https://aws.amazon.com/ec2/pricing/on-demand/) [↑](#footnote-ref-1)
2. This is a discounted price based on current demand, meaning that lesser-used instance types are offered at a discount. (https://aws.amazon.com/ec2/spot/pricing/) [↑](#footnote-ref-2)