

Deep Reinforcement Learning for Resource Management

by

Emma Wilson

B.Sc., Vertex University, 2019
M.Sc., The University of British Columbia, 2021

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA
(Vancouver)

August 2026

© Emma Wilson, 2026

The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, the dissertation entitled:

Deep Reinforcement Learning for Resource Management

submitted by Emma Wilson in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science.

Examining Committee:

Professor Laura Chen, Supervisor
Computer Science

Professor Omar Haddad, University Examiner
Electrical and Computer Engineering

Professor Mei Takahashi, University Examiner
Mathematics

Professor Rui Santos, External Examiner
Computer Science, Vertex University

Additional Supervisory Committee Members:

Professor Kwame Asante, Supervisory Committee Member
Computer Science

Abstract

Data centres allocate processors, memory and network bandwidth to thousands of jobs whose demands are uncertain and change over time. Hand-designed scheduling heuristics work well for the conditions their designers anticipated and poorly for the rest. This thesis studies reinforcement learning as a way to learn allocation policies from experience. It formulates cluster scheduling as a Markov decision process whose state encodes queued jobs and free resources, trains policies with a graph-structured network that generalizes across cluster sizes, and adds a constraint layer that guarantees no allocation exceeds capacity. In simulations driven by production traces, the learned scheduler reduces average job completion time by 19% relative to the best heuristic while never violating a capacity constraint.

Lay Summary

Computing centres run thousands of programs at once and must decide which program gets which machine, and when. The usual rules of thumb are written by hand and can make programs wait longer than they need to. This thesis trains a computer program to learn better decisions from practice runs, much as a person learns a game by playing it. The learned program finished work about a fifth faster than the best hand-written rules in our tests, and it was built so that it can never assign more work to a machine than the machine can hold.

Preface

This thesis is original, unpublished, independent work by the author. The simulator in Chapter 3 was built by the author on an open workload model; the production traces used in Chapter 4 are public and required no ethics approval.

Contents

Abstract	iii
Lay Summary	iv
Preface	v
Acknowledgements	ix
Dedication	x
1 Introduction	1
2 Background	2
3 A Constrained Graph Scheduler	3
4 Experiments	4
5 Conclusion	5
A Simulator and Hyperparameters	7

List of Tables

4.1	Average job completion time relative to shortest-job-first (lower is better). . . .	4
-----	---	---

List of Figures

3.1	The masked scheduling policy.	3
-----	---------------------------------------	---

Acknowledgements

I thank my supervisor, Professor Laura Chen, for her guidance and for the freedom to follow an idea into the weeds. I thank my supervisory committee for their advice, and my labmates for their friendship and their feedback on many practice talks.

Dedication

To Sam, for the dinners.

Chapter 1

Introduction

A cluster scheduler decides, many times per second, which queued job to start and on which machines. Good decisions shorten how long jobs wait; poor ones leave machines idle while work queues up. Production schedulers rely on heuristics such as shortest-job-first or dominant resource fairness, which are simple to reason about but cannot adapt to a particular workload.

Deep reinforcement learning has learned control policies for games [3] and early work showed it could learn simple packing policies [1]. This thesis asks whether it can produce schedulers that are both better than heuristics and safe to deploy.

Chapter 2

Background

A Markov decision process is a tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$. A policy $\pi_\theta(a \mid s)$ is trained to maximize the expected discounted return

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right], \quad (2.1)$$

here by proximal policy optimization [4]. For scheduling, the reward at each step is the negative number of jobs in the system, so maximizing J minimizes average job completion time by Little's law [5].

Chapter 3

A Constrained Graph Scheduler

We encode the cluster as a bipartite graph between queued jobs and machines, and a graph neural network produces a score for every feasible job-machine pair, following the design of Mao et al. [2]. A constraint layer masks any pair that would exceed a machine’s capacity before the softmax, so infeasible actions have probability zero rather than being penalized after the fact.

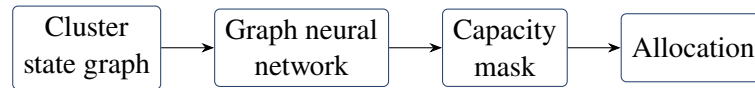


Figure 3.1: The masked scheduling policy.

Chapter 4

Experiments

We replayed four weeks of public cluster traces through the simulator. Table 4.1 reports average job completion time relative to shortest-job-first.

Table 4.1: Average job completion time relative to shortest-job-first (lower is better).

Scheduler	Relative completion time	Capacity violations
Shortest-job-first	1.00	0
Dominant resource fairness	1.07	0
Learned, unmasked	0.79	312
Learned, masked (this work)	0.81	0

The unmasked policy is slightly faster only because it overcommits machines; the masked policy keeps almost all of the gain with no violations.

Chapter 5

Conclusion

Learned schedulers can beat hand-written heuristics on realistic workloads, and masking infeasible actions makes them safe by construction. Transferring a policy trained in simulation to a live cluster, where the workload drifts, remains the main open problem.

Bibliography

- [1] Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. Resource management with deep reinforcement learning. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, pages 50–56, 2016.
- [2] Hongzi Mao, Malte Schwarzkopf, Shaileshh Bojja Venkatakrishnan, Zili Meng, and Mohammad Alizadeh. Learning scheduling algorithms for data processing clusters. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 270–288, 2019.
- [3] Volodymyr Mnih et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.
- [4] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv:1707.06347, 2017.
- [5] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, second edition, 2018.

Appendix A

Simulator and Hyperparameters

The simulator models machines with processor and memory capacities and jobs with stochastic durations drawn from the traces. Policies were trained for 2×10^7 environment steps with a discount factor $\gamma = 0.99$.