

ROYAL HOLLOWAY, UNIVERSITY OF LONDON

MSc THESIS

**Solving
the
Sliding Puzzle
&
Rubiks' Cube
by
Deep Reinforcement Learning**

Author:
Francois BERRIER

Supervisor:
Pr. Chris WATKINS

*A thesis submitted in fulfillment of the requirements
for the degree of MSc in Artificial Intelligence*

June 20, 2022

“The best advice I’ve ever received is ‘No one else knows what they’re doing either’”

Ricky Gervais

ROYAL HOLLOWAY, UNIVERSITY OF LONDON

Abstract

Computer Science Department

MSc in Artificial Intelligence

**Solving
the
Sliding Puzzle
&
Rubiks' Cube
by
Deep Reinforcement Learning**
by Francois BERRIER

blablabl...

Acknowledgements

I am grateful to my employer Bank of America for sponsoring my part time 2-year MSc in Artificial Intelligence at Royal Holloway. I entered in the study of AI with a heavy dose of skepticism and am delighted to have learnt so much and revised my judgment of this field. In particular I would like to thank my managers Mitrajit Dutta and Stephen Thompson for supporting me in pursuing this MSc.

I have been fortunate to have had the company of my colleague Saurabh Kumar, as he decided to follow me in this MSc journey. We have had countless opportunities to debate about all the new cool stuff we learnt during the MSc, and that has definitely made the whole process much more enjoyable.

I would also like to thank Professor Chris Watkins, inventor of QL :), for agreeing to supervise my project. It was an honour to attend his very lively lectures on Deep Learning during the first year of the MSc, as well as to absorb some of his wisdom and enthusiasm at our regular meetings during my work on this project.

Last but not least, my wife Thanh Nhã is my biggest supporter in everything I do, and her continued encouragement and enthusiasm in my pursuit of this MSc has made it so much easier to juggle between a demanding banking job, running 50 to 60 miles a week in preparation of multiple marathons and half marathons, and the coursework, lectures and exam revisions.

Contents

Abstract	v
Acknowledgements	vii
1 Objectives	1
1.1 Learning Objectives	1
1.2 Project's Objectives	2
2 Deep Reinforcement Learning Search	3
2.1 Reinforcement Learning	3
2.2 Deep Learning	3
2.3 Graph Search & Heuristics	3
2.4 Deep Reinforcement Learning Heuristics	3
2.5 Deep Q-Learning	3
3 Games	5
3.1 Sliding Puzzle	5
3.1.1 Search Space & Solvability	5
3.1.2 Optimal Cost	5
3.2 Rubiks' Cube	5
4 Code	7
4.1 Code organisation	7
4.2 Running An Example	7
4.3 DRL Training	7
4.4 Solvers Comparison	7
5 Sliding Puzzle	9
5.1 Low dimension	9
5.1.1 Perfect Heuristic	9
5.1.2 Results	9
5.2 3x4	9
5.3 4x4	9
6 Rubiks' Cube	11
6.1 2x2x2	11
6.2 3x3x3	11

List of Abbreviations

AIPnT	Artificial Intelligence Principles and Techniques
CS	Computer Science
CV	Computer Vision
DL	Deep Learning
DQL	Deep Q-Learning
DRL	Deep Reinforcement Learning
ML	Machine Learning
NLP	Natural Language Processing
RC	Rubiks' Cube
RL	Reinforcement Learning
RHUL	Royal Holloway, University of London
SP	Sliding Puzzle

Chapter 1

Objectives

1.1 Learning Objectives

Back when I studied Financial Mathematics, almost 2 decades ago, it was all about probability theory, stochastic calculus and asset (in particular derivatives) pricing. These skills were of course very sought after in the field of options trading, but were also often enough to get a job in algorithmic or systematic trading. By the middle of the 2010s, with the constant advances in computing power and storage, the better availability of off-the-shelves libraries and data sets, I witnessed a first revolution: the field of machine learning became more and more prominent and pretty much overshadowed other (more traditional maths) skills. More recently, a second revolution has taken not only the world of finance, but that of pretty much every science and industry, by storm: we are now in the artificial intelligence age. In 2019-2020, I decided it was time to see by myself what this was all about, and if the hype was justified. What better way to do that than embark on a proper MSc in Artificial Intelligence?

Of all the modules I have studied over the last two years of the Royal Holloway MSc in AI, I have been the most impressed by DL and NLP (itself arguably largely an application of DL) and particularly interested in AIPnT, especially our excursion in the field of graphs search (a very traditional CS topic, but which somehow I had not yet had a chance to study in much details). Even though I still believe there is a tremendous amount of malinvestment everywhere, due in good part to the inability of the average investor to distinguish between serious and scammy AI applications and startups (the same obviously goes for blockchain applications, which might warrant another MSc?), I have totally changed my mind around the potential of DL, DRL and NLP and think they are incredibly promising. I have been astonished to see by myself, through several of the courseworks we have done during the MSc, how incredibly efficient sophisticated ML, DL and DLR algorithms can be, when applied well on the right problems. Sometimes they just vastly outperform more naive and traditional approaches to the point of rendering older approaches entirely obsolete (e.g CV, NLP, game solvers, etc...).

For the project component of the MSc, I thought it would be interesting (and fun) for me to try and apply some of the DL, DRL and search techniques (from AIPnT) to a couple of single-player games, such as the sliding puzzle (of which some variations are well known under different names, e.g. the 8-puzzle and 15-puzzle) and of course the Rubik's cube. I am in particular looking to solidify my understanding of DRL by implementing and experimenting with concrete (though arguably of limited practical use) problems.

1.2 Project's Objectives

I am hoping with this project to implement and compare a few different methods to solve the Sliding Puzzle and the Rubik's Cube. Both these games have tremendously large state spaces (see section [Games](#) for details) and only one goal state. I am therefore likely to only succeed with reasonably small dimensional puzzles, especially since I have chosen for simplicity to implement things in Python. Depending on the progress I will be able to make in the imparted time, I am hoping to try a mix of simple searches (depth first search, breadth first search, A* with simple admissible heuristics), then more advanced ones such as A* informed by heuristics learnt via DL and DRL, as well as try different architectures and network sizes and designs for the DL and DRL heuristics. Time permitting I would like to give a go at DQL, and maybe also compare things with some open-source domain-specific implementations (for instance a Kociemba Rubik's algorithm implementation, see e.g. [Tsoy, 2019](#)).

Along the way, I am also hoping to learn a bit about these two games that I have chosen to work on, and maybe make a couple of remarks on them that the reader of this thesis might find interesting.

Chapter 2

Deep Reinforcement Learning Search

2.1 Reinforcement Learning

blabla

2.2 Deep Learning

blabla

2.3 Graph Search & Heuristics

See Dechter and Pearl, 1985

2.4 Deep Reinforcement Learning Heuristics

blabla

2.5 Deep Q-Learning

see Watkins and Dayan, 1992

Chapter 3

Games

3.1 Sliding Puzzle

The first puzzle I will focus on is the sliding puzzle (see Wikipedia, [2022](#)).

3.1.1 Search Space & Solvability

A proper proof of the solvability of half of the possible $n*m!/2$ puzzles of dimension n by m can be found in Archer, [1999](#)

3.1.2 Optimal Cost

blabla

3.2 Rubiks' Cube

blabla

Chapter 4

Code

4.1 Code organisation

The code I have developed for this project is all publicly available on my github page (Berrier, 2022)

4.2 Running An Example

blabla

4.3 DRL Training

blabla

4.4 Solvers Comparison

blabla]

Chapter 5

Sliding Puzzle

5.1 Low dimension

blablabla

5.1.1 Perfect Heuristic

blablabla

5.1.2 Results

blablabla

5.2 3x4

blabla

5.3 4x4

blabla

Chapter 6

Rubiks' Cube

6.1 2x2x2

blablabla

6.2 3x3x3

blabla

Bibliography

Journal Papers

- Archer, Aaron (1999). “A modern treatment of the 15 puzzle”. In: *American Mathematical Monthly* 106, pp. 793–799. URL: <https://www.cs.cmu.edu/afs/cs/academic/class/15859-f01/www/notes/15-puzzle.pdf>.
- Dechter, Rina and Judea Pearl (1985). “Generalized Best-First Search Strategies and the Optimality of A*”. In: *J. ACM* 32.3, pp. 505–536. DOI: [10.1145/3828.3830](https://doi.org/10.1145/3828.3830). URL: <https://doi.org/10.1145/3828.3830>.
- Watkins, Christopher J. C. H. and Peter Dayan (May 1992). “Q-learning”. In: *Machine Learning* 8.3, pp. 279–292. ISSN: 1573-0565. DOI: [10.1007/BF00992698](https://doi.org/10.1007/BF00992698). URL: <https://doi.org/10.1007/BF00992698>.

Misc

- Berrier (2022). *FB Code Base*. Accessed: 2022-06-22.
- Tsoy (2019). *Python Kociemba Solver*. [kociemba](#). Accessed: 2022-06-22.
- Wikipedia (2022). *Sliding Puzzle*. [wikipedia](#). Accessed: 2022-06-22.