CAPSTONE PROJECT 1
Planning Document

Evaluation of Nature-inspired Optimisation Algorithms in Solving Versus Tetris

by

Yap Wei Xiang
21067939

Supervisor: Dr Richard Wong Teck Ken

Semester: April 2024
Date:

Department of Computing and Information Systems
School of Engineering and Technology
Sunway University

Abstract

Contents

Abstract **i**

List of Tables **iii**

List of Figures **iv**

1 Introduction **1**
1.1 Motivation 1
1.2 Problem Statement 2
1.3 Aim 3
1.4 Objectives 3
1.5 Project Scope 3

2 Literature Review **4**
2.1 NP-completeness 4
2.1.1 Complexity Classes 4
2.1.2 Reductions 6
2.1.3 Tetris 6
2.2 Addressing NP-complete Problems 6
2.3 Nature-inspired Algorithms 6
2.4 Playing Tetris with Nature-inspired Algorithms 6
2.5 Identifying the Research Gaps 6
2.6 Concluding the Review 6

3 Technical Plan **7**

4 Work Plan **8**

5 References **9**

List of Tables

2.1 Comparison between (polynomial) n^2 and (exponential) 2^n as input size, n increases 5

List of Figures

1.1 A typical modern Tetris game where four lines are about to be cleared. The Tetrimino on the left of the matrix is the *Hold* piece and the pieces to the right of the matrix are collectively known as the *Queue*. 1

1.2 A typical game of Versus Tetris. Both players are trying to send lines to each other. The grey blocks are *Garbage Lines* sent from Player 2 (right) to Player 1 (left). 2

2.1 Visualisation of the sets P, NP, NP-hard and NP-complete. 5

1 Introduction

Tetris is a popular video game created in 1984 by computer programmer Alexey Pajitnov [1]. It is a puzzle game that requires players to strategically place sequences of pieces known as "Tetriminos" into a rectangular Matrix (refer to Figure 1.1). In the classic game, players attempt to clear as many lines as possible by completely filling horizontal rows of blocks, but if the Tetriminos surpass the top of the Matrix, the game ends.

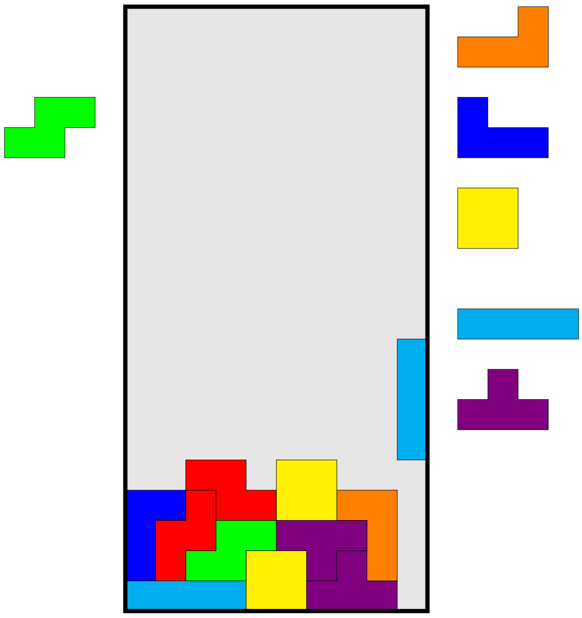

Figure 1.1: A typical modern Tetris game where four lines are about to be cleared. The Tetrimino on the left of the matrix is the *Hold* piece and the pieces to the right of the matrix are collectively known as the *Queue*.

Since its release, mathematicians and computer scientists have been intrigued by the game of Tetris, leading to a diverse array of research endeavours exploring the various facets of the game, including its computational complexity [2], and its possibility of being won [3] [4].

1.1 Motivation

In their paper, Demaine, Hohenberger, and Liben-Nowell showed that it is NP-complete to optimise several natural objective functions of Tetris [2]. NP-completeness poses a significant challenge in computational problem-solving, as it denotes the absence of polynomial-time algorithms for efficient solutions [5]. Moreover, the discovery of a polynomial-time algorithm for any NP-complete problem implies that any problem in the set of NP, encompassing efficiently verifiable but potentially difficult problems, could be solved in

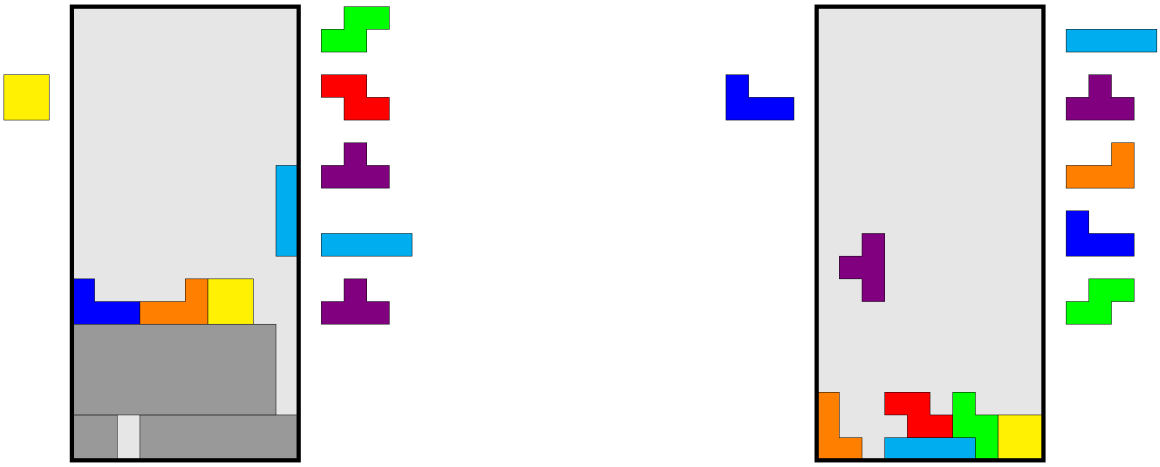

Figure 1.2: A typical game of Versus Tetris. Both players are trying to send lines to each other. The grey blocks are *Garbage Lines* sent from Player 2 (right) to Player 1 (left).

polynomial time [5]. NP-completeness extends beyond Tetris, with real-life instances of NP-complete arising in diverse fields such as route optimisation [6], job scheduling [7], and medicine [8].

To address these challenges, researchers have explored alternative approaches to tackle NP-complete problems, including the use of nature-inspired algorithms [9]. Although they might fail at finding optimal solutions, nature-inspired algorithms are able to return acceptable solutions in shorter running times [10]. In the context of optimising Tetris gameplay, studies have shown the effectiveness of using nature-inspired algorithms in playing the classic single-player game [11] [12]. However, there remains limited research on the effectiveness of nature-inspired optimisation algorithms in the multiplayer versus variant of the game.

1.2 Problem Statement

Versus Tetris (refer to Figure 1.2) presents a unique challenge in computational gaming due to its complex dynamics and real-time competitive nature. While previous research regarding the use of nature-inspired algorithms for Tetris optimisation have focused on single-player scenarios, the effectiveness of these algorithms in the multiplayer context remains largely unexplored. Despite the demonstrated success of these algorithms in improving single-player Tetris gameplay, their application to the multiplayer variant poses distinct challenges due to a different rule set and differing objectives that require further investigation.

1.3 Aim

The aim of this capstone project is to assess the effectiveness of nature-inspired optimisation algorithms in solving the game of Versus Tetris. By integrating insights from nature-inspired algorithms, the project seeks to create a robust and adaptable Tetris-playing software capable of competing against human players or other Tetris-playing programs. Through this endeavour, the project aims to contribute valuable insights into the application of nature-inspired algorithms in addressing computationally complex problems.

1.4 Objectives

The objectives of this project are as follows:

1. Formulate the problem of Versus Tetris for game AI.
2. Research and implement a variety of nature-inspired optimisation algorithms to determine their suitability for optimising gameplay strategies in Versus Tetris.
3. Design a comprehensive framework for objectively evaluating and comparing the performance of the algorithms.
4. Develop a playable game of Tetris that simulates gameplay and training.
5. Using the game, do comparative analyses with the designed framework to assess the effectiveness and efficiency of each algorithms.
6. Summarize findings from the comparative analyses.
7. Share the software with Tetris players of varying aptitudes to find the level of play for each algorithm.

1.5 Project Scope

This project will focus specifically on the evaluation of nature-inspired optimisation algorithms in the context of multiplayer versus Tetris. It will entail the development of a playable Tetris game capable of simulating gameplay and the training of algorithms. This simulation environment will facilitate in the analysis and evaluation of these algorithms' performances. The scope includes the exploring of a range of nature-inspired algorithms to address the unique challenges inherent in Versus Tetris.

2 Literature Review

2.1 NP-completeness

2.1.1 Complexity Classes

In computational complexity theory, complexity classes are sets of computational problems that are defined by fixing three parameters [13]:

1. The type of computational problem
2. The model of computation
3. The complexity measure and limiting function

The complexity class is then defined as the set of all problems that can be solved by the appropriate computational model, and such that for any input of a problem, the machine expends at most $f(|x|)$ units of a specified resource, x [14].

Before proceeding further, let us fix two of the three parameters. Firstly, we will adopt time as the measure of complexity. Secondly, we will exclusively focus on decision problems. Most common complexity classes refer to decision problems [15]. As described by Goldreich, these computational tasks involve determining if a given instance belongs in some predefined set [15].

Here, we introduce our first complexity class: *P*. *P* is defined by Sipser as the class of problems that are decidable in polynomial time on a deterministic single-tape Turing machine [5]. For this complexity class, the model of computation is deterministic.[1]

The complexity class *NP* is defined by Garey and Johnson as the class of all decision problems that can be solved in polynomial time by a nondeterministic algorithm [16]. The model of computation in this class is nondeterministic, which is an unrealistic model of computation [14]. This is because nondeterminism refers to the fact that a solution can be guessed out of polynomially many options in O(1) time [17]. Another characteristic of the *NP* class is that its solution can be verified in deterministic polynomial time [5].

Another complexity class to take note of is *NP-hard.* A problem is *NP-hard* if all problems in NP are polynomial time reducible to it [5]. A problem that is both *NP* and *NP-hard* is called *NP-complete* (refer to Figure 2.1) [17].

Since all problems in *NP* can be reduced to any *NP-complete* problem, if a deter-

[1]Deterministic refers to the fact that for any given input x, the machine's computation can proceed in exactly one way [5].

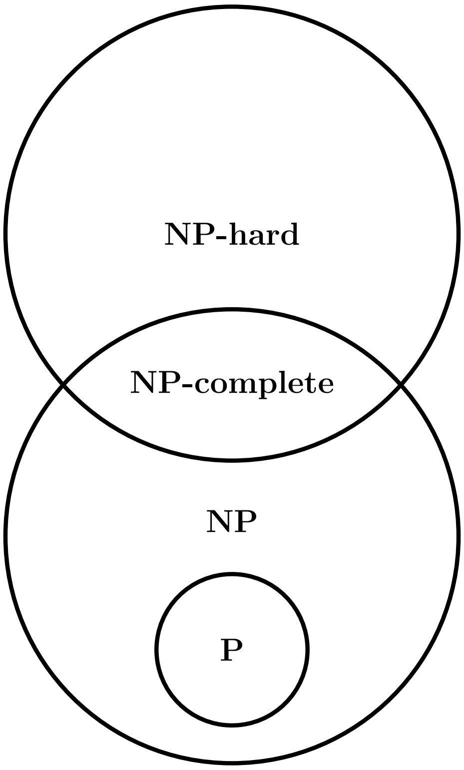

Figure 2.1: Visualisation of the sets P, NP, NP-hard and NP-complete.

ministic polynomial time algorithm can be found for an *NP-complete* problem, it would also mean that the set *NP* and *P* are equal sets. This problem was famously immortalised as one of the millennium prize problems from the Clay Institute of Mathematics [18].

Polynomial-time algorithms are defined as algorithms whose time complexity is $O(p(n))$ for some polynomial function, p, where n is the input size [16]. Any algorithm that cannot be so bounded is known as an exponential-time algorithm [16]. The distinction between these two algorithms become much more apparent as input sizes increase [16]. Table 2.1 shows why polynomial-time algorithms are more desirable when compared to exponential-time algorithms.

Table 2.1: Comparison between (polynomial) n^2 and (exponential) 2^n as input size, n increases

Time Complexity	**10**	**20**	**30**	**40**	**50**	**60**
n^2	.0001 seconds	.0004 seconds	.0009 seconds	.0016 seconds	.0025 seconds	.0036 seconds
2^n	.001 seconds	1.0 seconds	17.9 minutes	12.7 days	35.7 years	366 centuries

2.1.2 Reductions

2.1.3 Tetris

2.2 Addressing NP-complete Problems

2.3 Nature-inspired Algorithms

2.4 Playing Tetris with Nature-inspired Algorithms

2.5 Identifying the Research Gaps

2.6 Concluding the Review

3 Technical Plan

4 Work Plan

5 References

[1] Tetris Inc., *About Tetris*, https://tetris.com/about-us, [accessed Apr. 22, 2024].

[2] E. D. Demaine, S. Hohenberger, and D. Liben-Nowell, "Tetris is hard, even to approximate," in *Computing and Combinatorics*, T. Warnow and B. Zhu, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 351–363, ISBN: 978-3-540-45071-9.

[3] J. Brzustowski, "Can you win at tetris?" Master's Thesis, University of Waterloo, 200 University Ave W, Waterloo, ON N2L 3G1, Canada, 1988.

[4] H. Burgiel, "How to lose at tetris," *The Mathematical Gazette*, vol. 81, no. 491, pp. 194–200, 1997. DOI: 10.2307/3619195.

[5] M. Sipser, in *Introduction to the Theory of Computation*, Cengage Learning, 2013.

[6] V. Lesch, M. König, S. Kounev, A. Stein, and C. Krupitzer, "A case study of vehicle route optimization," *CoRR*, vol. abs/2111.09087, 2021.

[7] J. D. Ullman, "Np-complete scheduling problems," *Journal of Computer and System sciences*, vol. 10, no. 3, pp. 384–393, 1975.

[8] J. Arle and K. Carlson, "Medical diagnosis and treatment is np-complete," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 33, pp. 1–16, Mar. 2020. DOI: 10.1080/0952813X.2020.1737581.

[9] L. Davis, "Job shop scheduling with genetic algorithms," in *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, 1985, pp. 136–140.

[10] W. Korani and M. Mouhoub, "Review on nature-inspired algorithms," *Operations Research Forum*, vol. 2, Jul. 2021. DOI: 10.1007/s43069-021-00068-x.

[11] J. Lewis, "Playing tetris with genetic algorithms," 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:17416568.

[12] L. Langenhoven, W. S. van Heerden, and A. P. Engelbrecht, "Swarm tetris: Applying particle swarm optimization to tetris," in *IEEE Congress on Evolutionary Computation*, 2010, pp. 1–8. DOI: 10.1109/CEC.2010.5586033.

[13] O. Goldreich, "Computational complexity: A conceptual perspective," *SIGACT News*, vol. 39, no. 3, pp. 35–39, Sep. 2008, ISSN: 0163-5700. DOI: 10.1145/1412700.1412710. [Online]. Available: https://doi.org/10.1145/1412700.1412710.

[14] C. H. Papadimitriou, *Computational complexity.* Addison-Wesley, 1994, pp. I–XV, 1–523, ISBN: 978-0-201-53082-7.

[15] O. Goldreich, *P, NP, and NP-Completeness: The Basics of Computational Complexity*. Cambridge University Press, 2010.

[16] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, First Edition. W. H. Freeman, 1979, ISBN: 0716710455.

[17] MIT OpenCourseWare. "Np-completeness." 6.046J: Design and Analysis on Algorithms, Massachusetts Institute of Technology. (2015), [Online]. Available: https://ocw.mit.edu/courses/6-046j-design-and-analysis-of-algorithms-spring-2015/resources/mit6_046js15_lec16/.

[18] *The Millennium Prize Problems - Clay Mathematics Institute — claymath.org*, https://www.claymath.org/millennium-problems/, [Accessed 15-05-2024].