Analyzing Strategies for Voronoi Area Game CS460-2024 Group Project: Midway Report

Samir Dileep
School of Biological Sciences
National Institute of Science Education and Research
Jatni, Odisha-752050

Sandipan Samanta
School of Mathematical Sciences
National Institute of Science Education and Research
Jatni, Odisha-752050

Abstract

This project aims to design and train a Machine Learning model to master the 2-player Voronoi Area Game in 2D. An in-house replica of the game was made in python and the training process was run on this interface. After training, we expect the model to make the most optimal placement of points to win an instance of the game when placing as Red or as Blue. The model also helps answer which of Red and Blue have the advantage when both play optimally.

1 Introduction

1.1 Voronoi Diagrams

A Voronoi diagram is a method of geometrical partitioning of a space where the division is based on the distance of every point in the space from a specific set of points. The distance metric used here is Euclidean, but any other metric can be used, depending on the application.

Let X be a metric space with a distance metric d. Let $P = \{P_1, P_2, ..., P_n\}$ be the set of points to be considered as sites based on which the division is done. Then, the region R_i assigned to a point $P_i \in P$ will be given by:

$$R_i = \{x \in X \mid d(x, P_i) \leq d(x, P_j) \,\forall\, P_j \in P;\; P_i \neq P_j\} \tag{1}$$

Put simply, the Voronoi diagram can be defined as the collection of R_i from all P_i in P.

1.2 The Voronoi Area Game in 2D

The Voronoi area game (hereafter referred to as the **Voronoi game**) is one where a finite space is divided according to a Voronoi diagram and the winner is decided based on which subset of points cover the higher proportion of area. Players take turns to add points to a finite space which will act as sites for the Voronoi diagram. For this project, we consider only the two-player variant in 2D with 5 turns; complex variants having more players in higher dimensions with more turns may be studied with some variations in the code, although the computational power required for the same will exponentially increase.

The steps involved in an instance of the Voronoi game with two players (**Red** and **Blue**) are given below. Without loss of generality, we take Red to make the first turn in every game.

1. Red places its point in the given limited playing space. The corresponding Voronoi diagram is obtained, where each region is coloured with that of its site. Here, all of the area would be assigned to Red. The proportion of area covered by each player can be obtained at every turn.
2. Blue places its point in the space such that it is at least a certain unit of distance d away from the point. Once again, the corresponding Voronoi diagram is made, and we see both red and blue regions along with the proportion of the total area they cover.
3. Both players keep adding their points to the space such that every point is at least d units away from each other till 5 turns are complete. The player covering the most area wins.

1.3 Optimal Strategies using ML

In this project, our aim is to create an ML model which would make the most optimal placement of points to win after 5 turns as either Red or Blue. For this, two versions of the model is to be made, each for playing as Red and Blue respectively.

Through applications of this model, we hope to answer some fundamental questions about the game, some of which are:

1. **Do any of the players hold an automatic advantage?** We plan to answer this by making the two fully trained models play against each other multiple times.
2. **Is a greedy approach the most optimal strategy?** Here, greedy refers to a player placing a point such that the highest possible area is secured for the current turn. Comparing a greedy placement of points with that of the trained models, both against the same placement of opposite points should give a clear answer to this question.

2 Execution of Midway Goals

Two goals were planned to be completed by the midway period. One was to **engineer an interface** on which the model would play multiple iterations of the Voronoi game during the learning process. The second goal was to start **implementation of ML algorithms** on the interface.

2.1 The game interface

The interface models a simplified version of the Voronoi game where a 100×100 grid is taken as the playing area. Points can be plotted with at least 10 units of distance between them. The code gives the set of sites in the area as well as the proportion of area covered by both players for all turns. It also shows the final Voronoi diagram which determines the winner. The python code for the interface can be found here.

2.2 ML Implementation

Unfortunately, we were not able to start implementing an ML algorithm for the project yet. We planned to use Q-learning for the learning process, but the Q-matrix involved turned out to be too large to feasibly work with, even for a 100×100 grid. This was due to the fact that placements of points in every previous turn have to be included in the Q-matrix for one single turn. Hence, we have to change the state representation so that decisions can be taken in a reasonable amount of time.

3 Endterm Goals

The primary focus for the endterm will be on developing an effective state representation and employing machine learning approaches, including Convolutional Neural Networks (CNNs) and potentially other algorithms, to train models capable of playing the Voronoi Area Game optimally as both Red and Blue players. The specific goals are as follows:

3.1 State Representation

Devise a compact and efficient state representation that encapsulates the relevant information from the game board and prior moves, while ensuring the state space remains manageable for the machine learning algorithms. Techniques such as dimensionality reduction or encoding schemes may be explored for this purpose.

3.2 CNN Training

1. Generate a fixed number of games, say 100, with random point placements to serve as the initial training dataset for the CNN model.
2. Adopt an approach similar to reinforcement learning, where instead of providing rewards for individual states and actions, the CNN model will analyze the image-like dataset of game boards and provide rewards to the system based on its analysis.
3. Train the CNN model on this dataset to learn strategies for playing the Voronoi Area Game.
4. Make the trained CNN model play against itself for an additional set of games, e.g. 100 games.
5. Feed the new set of self-played games back into the training process to further improve the model's performance.
6. Repeat steps 4 and 5 for several iterations until the model converges to an optimal or near-optimal strategy.

3.3 Other Algorithms

Explore the implementation of other machine learning algorithms, in addition to CNNs, to train models for the Voronoi Area Game. This will allow for a comparative analysis of different approaches and their effectiveness in learning optimal strategies.

3.4 Model Evaluation

Evaluate the trained models (CNN and other algorithms) by pitting them against greedy and random play algorithms, which have already been implemented. Optionally, arrange for the trained models to play against human players to assess their performance against human intuition and strategies.

3.5 Strategy Analysis

Analyze the strategies learned by the trained models to gain insights into optimal play in the Voronoi Area Game. This may involve visualizing the learned policies, identifying key decision points, and comparing the strategies to existing theoretical results or human intuition.

3.6 Documentation and Report

Thoroughly document the project's methodology, implementation details, results, findings, and any insights gained from the strategy analysis in a comprehensive final report.

Through the accomplishment of these goals, we aim to develop a deep understanding of the Voronoi Area Game's strategic landscape and contribute to the field of game theory and machine learning applications in game-playing scenarios.