**Abstract**

In our experiment, we are using total 8 features to drive our agent. Namely, they are: Landing Height, Eroded Piece Cells, Row Transitions, Column Transitions, Number of Holes, Board Wells, Number of Rows with Holes, and Hole Depth. In order to make the features useful, we need to find the appropriate weight vector. To this end, we employed the use of a couple of well-known optimizers, specifically, Harmony Optimization and Genetic Optimization, to find these weights.

**Features**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Description** | **Reason** |
| Landing height | The height where the last piece is added into the board before any full row is cleared. (the row index of the topmost cell of the landed piece) | Punishes increasing pile heights. |
| Eroded piece cells | (Number of rows cleared after landing the piece) x (Number of eliminated cells of the landed piece). | Encourages clearing rows. |
| Row transitions | The number of horizontal transitions from occupied to unoccupied or vice versa on the board. (the left and right exteriors are considered occupied) | Discourages creating of horizontal heterogeneous rows. |
| Column transitions | Same ideas as row transitions. (the bottom and top exteriors are considered occupied and unoccupied respectively) | Discourages creating of vertical heterogeneous columns. |
| Holes | The number of unoccupied cells covered by at least one occupied cell above. | Punishes creating of holes |
| Board wells | ∑w∈wells(1 + 2 + … + depth(w)). | Discourages making deep wells1 because they are harder to clear. |
| Row holes | The number of rows having at least one hole. | Discourages creating row holes. |
| Hole depth | Sum of the value of each hole where the value of a hole is the number of occupied cells above it. | Discourages placing of pieces above a column with hole(s). |

1 A well is a succession of unoccupied cells whose left cells and right cells are both occupied.

Because the evaluation function to pick the best move for each turn is a linear combination, all the points of the 8-dimensional search space of weight vectors that fall on the same line passing through the origin have the same evaluated value. In order to adapt to this characteristic, we limit the search space of weight vectors to the surface of the 8-dimensional unit sphere. This means we need to normalize weight vectors.

As some features are meant to reinforce various actions whilst others discourages, hence they are assigned positive and negative weights respectively.

After applying the 2 observations above to the search space of weight vectors, we can minimize the search space, thus increasing the convergence rate of weight vectors. The table below displays the range for each feature weight:

|  |  |
| --- | --- |
|  | **Range** |
| One positive feature: eroded piece cells. | [0.0, 1.0] |
| The remaining features are negative. | [-1.0, 0.0] |

Due to the random nature of Tetris, we have to test a weight vector in an adequate number of games in order to obtain an accurate fitness value – this is done by taking the average.

Additionally, in order to expedite the weight vector optimization procedure, we simulate the game on a “hard” version of Tetris which increases the convergence rate of weight vectors. The only difference between the “hard” version and the normal one is the frequency at which each piece is produced. In the “hard” version, the probabilities of S and Z pieces are 3/11, whilst the probabilities for the rest are 1/11.

**Experiment Analysis**

In order to measure the performance of weight vectors, we introduce the term the ratio of spawned pieces (SP) / cleared rows (CR). Theoretically, given 10-cells rows and 4-cells pieces, the best case for Tetris is to be able to clear one row using 2.5 spawned pieces on average. Hence, the lower bound for the ratio of spawned pieces / cleared rows is 2.5.

We consider the two different weight vectors having different SP/CR ratio, α and β. If the two weight vectors are evaluated with the same number of spawned pieces and α < β, the number of cleared rows of the α-weight vector is more than that of the β-weight vectors. In other words, the nearer the ratio SP/CR of a weight vector to 2.5 is, the better its performance is in terms of the number of cleared rows.

In our experiment, we make an assumption that using “hard” case to perform optimization process provides a lower bound for the “Normal” case and speed up our experiment.

For our experimental protocol, we use the best weight vectors found by the “hard” version, as shown in Table 1 and Table 3, of Tetris to play on the normal version, as shown in Table 2 and Table 4. From our collected data, the ratios SP/CR of both “hard” and normal versions converge at values very near to the optimal point, 2.5. This shows that our best weight vectors perform well. Additionally, the evidences that the convergence rate and the number of cleared rows in the normal version are better than those in the “hard” version support our assumption of using the “hard” version to approximate the normal version.

|  |  |  |  |
| --- | --- | --- | --- |
| Genetic Approach Hard Case | | | |
| Weight Set Number | Average1 Row cleared | Average1 Piece Spawn | Average Ratio2 |
| 1 | 870 | 2207.12 | 2.53691954 |
| 2 | 798.42 | 2028.68 | 2.540868215 |
| 3 | 752.24 | 1912.68 | 2.542645964 |
| 4 | 732.9 | 1863.68 | 2.542884432 |

Table 1

|  |  |  |  |
| --- | --- | --- | --- |
| Genetic Approach Normal Case | | | |
| Weight Set Number | Average3 Row Cleared | Average3 Piece Spawn | Average Ratio2 |
| 1 | 487200 | 1218030.7 | 2.500063013 |
| 2 | 389329.7 | 973356.3 | 2.500082321 |
| 4 | 293160.2 | 732931.5 | 2.500105744 |
| 3 | 210670.6 | 526708.5 | 2.500151896 |

Table 2

|  |  |  |  |
| --- | --- | --- | --- |
| Harmony Approach Hard Case | | | |
| Weight Set Number | Average[[1]](#footnote-1) RowCleared | Average1 Piece Spawn | Average[[2]](#footnote-2) Ratio |
| 1 | 962.38 | 2439.32 | 2.534674453 |
| 2 | 820.4 | 2083.26 | 2.539322282 |
| 3 | 824.64 | 2094.44 | 2.539823438 |
| 4 | 788.28 | 2002.64 | 2.540518597 |
| 5 | 772.68 | 1964.4 | 2.542320236 |

Table 3

|  |  |  |  |
| --- | --- | --- | --- |
| Harmony Approach Normal Case | | | |
| Weight Set Number | Average[[3]](#footnote-3) Row Cleared | Average3 Piece Spawn | Average2 Ratio |
| 4 | 651838 | 1629629 | 2.500052160 |
| 3 | 532657 | 1331674 | 2.500059137 |
| 2 | 518466 | 1296196 | 2.500059792 |
| 1 | 471700 | 1179282 | 2.500067840 |
| 5 | 366425 | 916096 | 2.500091424 |

Table 4

From the two approaches we employ in this experiment, we get the best average ratio 2.500063013 and 2.500052160 for genetic and harmony approach respectively. Given these two ratio, we can conclude[[4]](#footnote-4) that the weight set we get from harmony approach is perform better than the genetic approach. The best weight set we obtained is {-0.37550838569759515, 0.2330609762288089, -0.30192169246015765, -0.5978769335300016, -0.19864323164772435, -0.3133256472174362, -0.03688152862554421, -0.4659078833496842}. From statistical point of view, the confidence for this weight set is low since the sample size is small.

1. We using 50 runs to get the averages for each weight set. [↑](#footnote-ref-1)
2. Average ratio is taken from the ratio of average piece spawn against average row cleared. Theoretically, the optimum average ratio is exactly 2.5 given that the size of Tetris with 10 columns and each piece contains exactly 4 cells. [↑](#footnote-ref-2)
3. We using 10 runs to get the averages for each weight set. [↑](#footnote-ref-3)
4. This conclusion is showing the best result up to the point of our experiment. [↑](#footnote-ref-4)