CS3243 Artificial Intelligence Term Project

Group Members :
Eng Teng Chuan (A0094593R)
Hisyam Nursaid Indrakesuma (A0098139R)
Daniel Dimas Dewangkara (A0083934X)
Steven Kester Yuwono (A0080415N)
Yohanes Lim (A0099768Y)

Introduction

This report will first introduce two strategies used by our agent with the fundamental algorithm being an existing solution to the Tetris problem. We will then display the results of running the tweaked algorithm against a test set on a chart, briefly mentioning the average and best number of lines cleared across all simulations. The report will end off with an analysis on the results and strategies used, as well as our group's novel additions.

Strategy

Introduction

We decided to use the heuristics and weights mentioned in El-Ashi's approach[1]. Six heuristics are mentioned in El-Ashi's approach, along with its related weight which indicates its importance in choosing the next move for the tetris tile. We then came up with two strategies to attempt to improve on the approach, as mentioned below.

One-Step Look Ahead

The algorithm performs a one-step look ahead to determine the best move to perform based on the tile received. The look ahead performs as follows:

1. For current tile, determine the three best spots tile could land on (C1, C2, C3)
2. "Look ahead" one step, where agent does not know which of the seven tiles will be next
3. For C1:
    a. Determine the best spot where each of the future tiles could land on (F1, F2, …, F7)
    b. Take worst of the set of scores {F1, F2, …, F7}
    c. Add this score to C1 (Resultant score = R1)
4. Repeat Step 3 for C2 and C3 to get R2 and R3
5. Take the best of R1, R2 and R3. Best score determines what move agent should choose between C1, C2 and C3.

Simply put, the agent determines the best spot the current tile could be placed, based on the worst possible tile it could receive for the next move.

New Weights from Genetic Algorithm

The second strategy used is the Genetic Algorithm[2]. We started with 21 sets of 6 weights and derived the best set of weights which would replace the original six heuristic weights. The modified weights are as shown below.

Heuristic	Description	Weight
Rows Eliminated	Number of rows eliminated	3.781057045291721
Number of Holes	Number of empty cells that have at least one filled cell above it in the same column	-8.533854039893987

[1] http://ielashi.com/el-tetris-an-improvement-on-pierre-dellacheries-algorithm/

[2] http://codemyroad.wordpress.com/2013/04/14/tetris-ai-the-near-perfect-player/

Number of Wells	A well is a succession of empty cells such that their left and right cells are both filled	-7.053276659088376
Number of Column Transitions	Column transitions: occurs when an empty cell is adjacent to a filled cell on the same column and vice versa	-7.708336085296365
Number of Row Transitions	Same as Column Transitions, but applying to rows	-2.5799680732117745
Landing Height	Height where piece is placed (column height + piece height/2)	-3.136943472726582

Table 1: New Weights for El-Ashi's Approach

Linear Rating Function vs Exponential Rating Function

We considered two approaches for our agent. The agent will utilize the linear rating function (dot product) $[R_1(b) = \sum_{i=1}^{n} \omega_i r_i(b)]$ or exponential rating-function $[R_e(b) = \sum_{i=1}^{n} \omega_i r_i(b)^{e_i}]$.

While linear rating function is straightforward in nature, it may not be useful in approximating a good tetris move. This is because from a human player's perspective, a short pile height is not as significant as that of a high pile height, which is close to a game over. However, the linear rating function sees any pile height equally, hence a discrepancy. Both functions were still tested, their weights obtained from the Genetic Algorithm. The weights utilized are as shown in *Table 2*.

Heuristic	**Weight**	**Related Exponent Weight**
Rows Eliminated	11.429954479098788	9.763232628810648
Number of Holes	-11.169138960465807	1.6993720935711176
Number of Wells	-10.55341320502331	5.333692367852307
Number of Column Transitions	-9.992527055551824	1.1603835054247467
Number of Row Transitions	-10.732376458867385	1.0796379646719207
Landing Height	-8.000390018764028	1.3696465899099395

Table 2: Rating Function weights

Results

On the initial test, 25 simulation runs were performed using a completely random selection of tiles. The average number of lines cleared under various conditions are listed as follows.

	Average number of lines cleared
Original Weights from El-Ashi's approach, no look ahead	~100,000
Genetic Algorithm weights, linear rating function, no look ahead	~120,000
Genetic Algorithm weights, linear rating function, with look ahead	~160,000
Genetic Algorithm weights, exponential rating function, with look ahead	~4,000

Table 3: Average Number of Lines Cleared (All shapes)

From the results above, we found that there was no difference whether the agent used a linear rating function or not, and it performed much worse with an exponential rating function. The next test was thus carried out with a linear rating function with look ahead. A total of 50 simulations were run. We tested our agent with the original El-Ashi's weights and with the new set of weights chosen by Genetic Algorithm. This test set involved only two tiles, the 'S' and 'Z' tiles, picked at random. The results of the simulations are as shown in the chart below. Several other statistics are shown after the chart, on the average and most number of lines cleared for each approach.

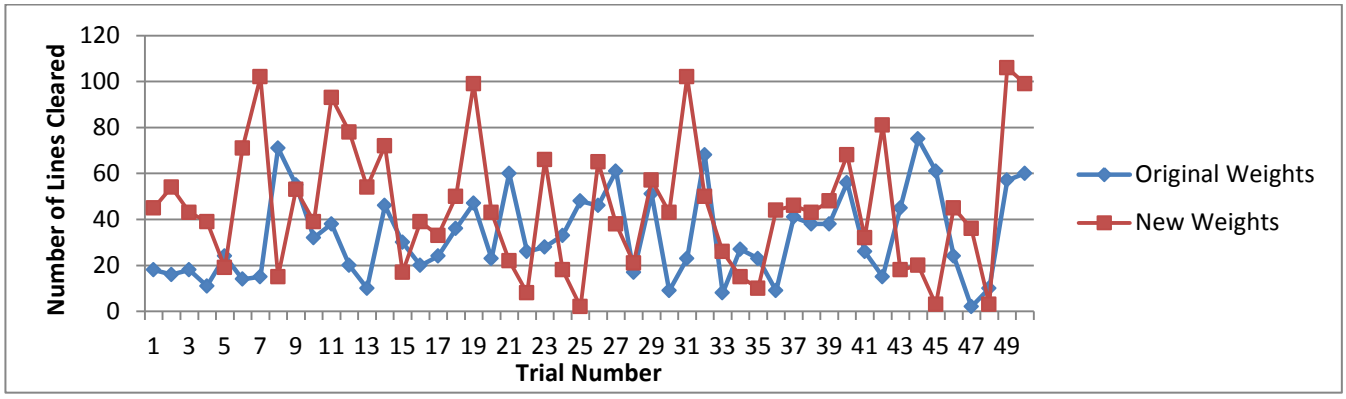

Figure 1: Completed number of lines based on weights

	Average number of lines cleared	**Most number of lines cleared**
Original Weights from El-Ashi's approach	33	75
Genetic Algorithm weights	45	106

Table 4: Lines Cleared ('S' and 'Z' shapes only)

Analysis

From *Table 3*, we can see that our weights perform marginally better than El-Ashi's. More importantly, we reaffirm that the number of holes and number of column transitions are the most dominant weights. The importance of this will be discussed in the conclusion.

We achieved even better results by utilizing the one-step look ahead method, which was to be expected, by avoiding worse scenarios from the possible moves that could be made. However, due to the need for more calculations, the time taken to determine the next generation of weights with look ahead is longer. It should be noted that the number of generations needed to converge to the answer are similar (there is no correlation between avoiding worse cases and faster convergence).

In our tests, the exponential rating function performed much worse compared to the others. This is probably because 6 weights with 6 exponential values prove to be too large of a search space with genetic algorithm. As a result, the algorithm gets trapped in the local maxima which are distributed across the search space easily. From *Table 3*, we see that the most dominant weight is Rows Eliminated, which is not congruent with what we had obtained before (Number of Holes). Moreover, it is too dominant with an exponent of 9 so we are sure that it is trapped in a local maximum.

Conclusion

Our novel and significant contribution is our approach of using combinatorial search in look ahead to determine the best set of weights to use. Many websites or papers tend to just use a single algorithm which we feel is too restrictive. However, we feel that utilizing combinatorial search helps to improve the result. This set of weights must of course, be obtained beforehand to save calculation time.

In addition, we find that when using genetic algorithm, it is advisable not to have too many weights. As shown in the exponential rating function, too many weights will result in more local maxima for the algorithm to be trapped in. We need to determine the ideal combination of weights beforehand instead of running the combination that we have, expecting it to get closer to the global maximum.

In conclusion, for future endeavors of this problem, we recommend to use a combinatorial search method on top of the algorithm that you will be using (and any other suitable methodology). Should they choose to use genetic algorithm, we recommend to experiment with different combinations of weights and not be fixated with a predetermined combination of weights.