Notes on tetris heuristics

<https://codemyroad.wordpress.com/2013/04/14/tetris-ai-the-near-perfect-player/>

A bot that can indefinitely clear lines in a single Tetris game – uses genetic algorithm!

* There are many types of tetris, which one are we using
* Make a set of the rules our one follows
* Aggregate height
  + How high a grid is
  + Sum of the height of each column
  + Value should be minimized
* Complete lines
  + Number of complete lines in the grid
  + Value should be maximized
* Holes
  + Empty space such that there is at least one tile in the same column above it
  + Value should be minimized
* Bumpiness
  + Variation of the grid’s column heights
  + Sum up the abs differences between all two adjacent columns
  + Value should be minimized

<https://luckytoilet.wordpress.com/2011/05/27/coding-a-tetris-ai-using-a-genetic-algorithm/>

Another genetic algorithm! Yay

* Blockades
  + Any block that’s immediately above a hole
  + Value should be minimized
* Hugging walls / floors, flattening (= current block edge touches existing block edge)

<http://totologic.blogspot.sg/2013/03/tetris-ai-explained.html>

* Highest full cell – minimize

This is the article Meriem found, so pass

* Having a ‘holes’ heuristic which makes an aggregate value taking into consideration different hole types and their rankings might be a good idea

((<http://meatfighter.com/nintendotetrisai/)>)

<https://web.engr.illinois.edu/~cyen4/pdf/Tetris_AI.pdf>

* Balancing the height of sides?
* Minimizing height difference between columns

<http://www.cs.cmu.edu/afs/cs/project/ACRL/www/TetrisReports/Breelyn_Eric_Don_Project.pdf>

* Group row clearance incentive – like a multiplier or something
* Once a hole is closed / covered – ignore it?
* Scoring gap fill points – more points earned when the match fills a smaller / harder to fill gap
* Implementing an overall game strategy?
  + Leaving columns open
  + ((<http://www.tetrisconcept.com/p/start.html>))

APRIL 20th MEETING – HEURISTIC NOTES

* 4 or 5 heuristics
  + Overfitting and balancing between training time and development and slowing computation – training each parameter
* 7 tetrominos – how to rotate, to be discussed later – taken into account during actions
* Completely classic tetris
* T-spins? lol
* Maximum height (1) VS aggregate height (2)
  + Position doesn’t matter
  + Can use both, just for maximum coverage for now
  + Minimising
* Bumpiness (3) – difference between adjacent column slopes (absolute)
  + Minimising
* Clearing rows (4) – best thing to happen
  + Multiplier for number of rows – more rows is better
  + Maximising
* Concavity – ‘wells’
  + Bumpiness heuristic gets rid of this problem
* Holes (5)
  + Hole = one space covered at the top
  + At the top level any missing spaces = gaps – every gap below is a hole, single unit
  + Difference between vertical and horizontal holes / gaps
  + Possibly ranked holes
  + Calculating gap costs
  + 10 possible gap types
* Implement the ones he said in the paper first
* Lookahead ? later
* Combining the features in CMU and project file
  + Reversing – making it negative
  + Minimising all the heuristics
* Weights
  + Limiting the weights?
  + The purpose of the weights – to enforce the importance difference between the heuristics
  + Limiting them or controlling the value doesn’t really effect the outcome
  + **Weight training??**
* Pruning heuristic – looking at best five options
* We reconfigure the board after each action
* Instantaneous cost VS state cost
  + 1 = function, returns the row you’ve cleared – binary value return, 0 if no rows cleared, 1 if row cleared
* Is the main goal maintaining a low height or clearing lines??
* Death (6) – MAXIMUM COST, instant no go – positive infinite
* The only file we can edit is PlayerSkeleton – the provided methods to understand the state are not very extensive
* Next piece is uniformly random
* Ummmm space invaders? LOL

Comments

* Each do one heuristic and create a standardized way to combine it – everyone makes a function of it that returns a value to an overarching heuristic