2048 Expectimax Solver

Anders Sildnes, Andrej Leitner *students*

***Abstract*—This text answers assignment 5: writing a solver for the game 2048. The purpose of the game is to slide tiles with numbers 2^n, and join them together to form 2048 (2^{11}), or possibly higher. We present a solver using Expectimax algorithm. We explain our choice of heuristics and results.**

2048 can be thought of as a 2-player, turn-based game. The opponent places tiles valued either 2 or 4 in an availible location. Then, the other player makes a move to slide all tiles in a given direction. This goes back and forth until there are no more availible moves.

In a turn-based game, you have the time to consider the consequences for each possible action. This gives computers immense advantages over humans: IBM's chess-solving machine "Deep Blue" won against Garry Kasparov in 1997 considered more than 200 million possible moves per second[1].

In the case of 2048, this could yield a valid solution in a short amount of time. However, not everyone has access to such fast hardware and multi-threading so a way to prune the search space is needed.

Minimax Algorithm

The purpose of minimax is to minimize loss for a worst possible scenario. This means choosing an action such that your opponent can do "least amount of damages", i.e. leaving you in a promising state. This is also called *adversial search*. For 2048 this would mean always making a move such that no matter where the next tile goes, and no matter what value, you will still find a way for success.

[1]For more, see http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/

2048, however, is a stochastic game. The opponent will never purposly select a bad tile, nor *choose* a bad value. This is to say that there is no opponent, so it is possible to always land a best case scenario. Therefore, we felt that minimax, considering the worst case, is too pessimistic in its search space. Furthermore, the distribution of numbered tiles are given: there is a 90% chance of getting a tile valued 2, and 10% chance of getting a tile valued 4. The location is random. Also note that in some cases, a tile in position i, j is no different from having a tile in position $i+i, j$ if the next move it to slide the tiles downward. Thus the possible search space is not too big and one can use statistics to get accurate enough resuls.

Using stochastic information in adversial search leads to expectimax. After expanding the search space up to a given depth, you can back up from each node and introduce a chance node. The chance nodes multiply the value to each of its children (given a state, the objective value to each of the children). The chance nodes are used when pruning the search space. Their importance is that unlikely values will yield a lower penalty to the overall score of the search node, such that the node is not necessarily pruned away.

Hey

end