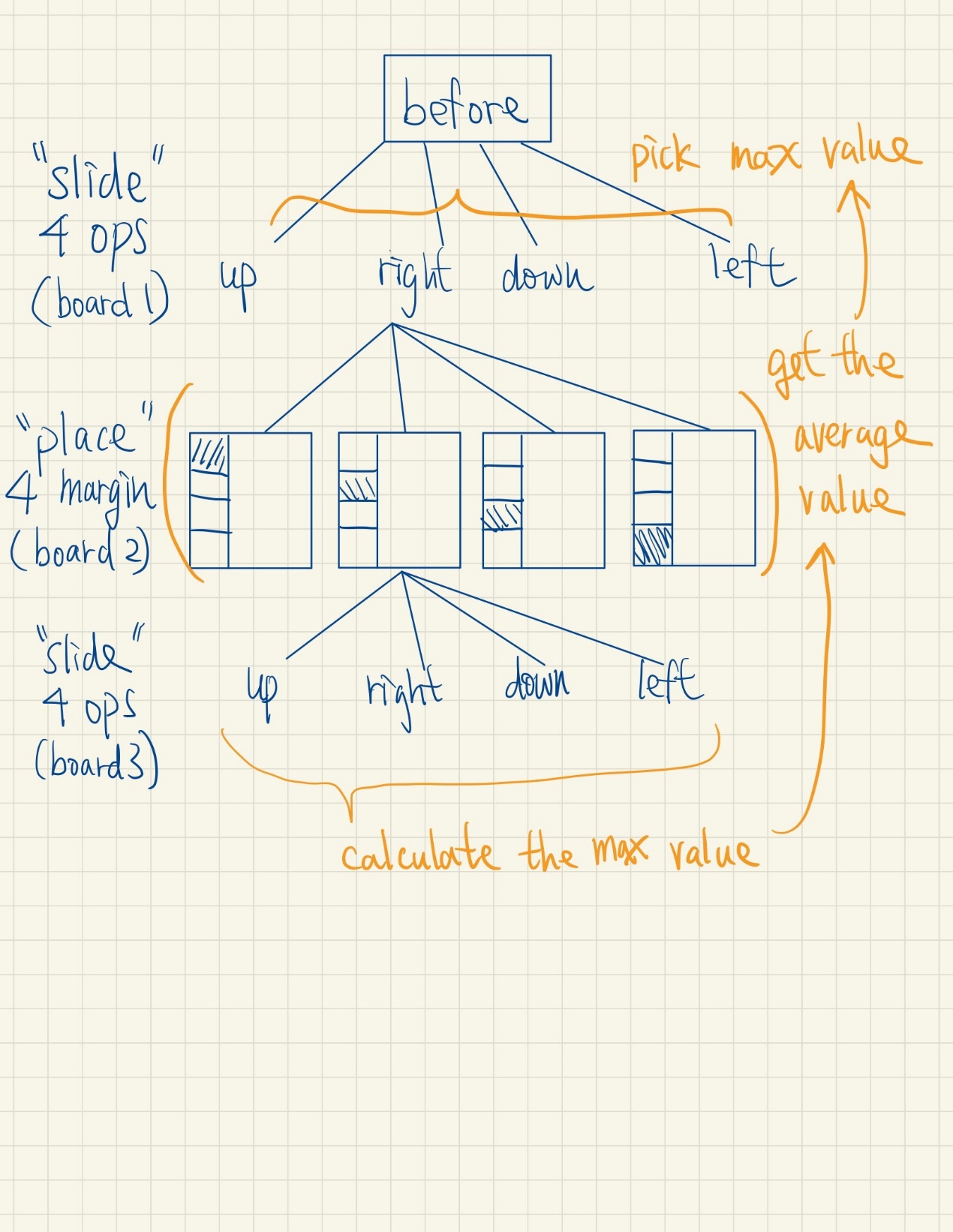
**Report of TCG Project 2+**

109550027紀竺均

1. Improvements
2. I add expectimax search on my TD-learning model. Below is how I implement expectimax function. To sum, it expand all possible next moves and calculate next moves’ expected value by place ‘hint’ and pretend to slide on the board.



1. Two-step TD learning:

Cause we have simulate “next” move and the move after next move (called it “next next” move here) from expectimax search function, I add the “next next” move’s reward and its board into consideration in my TD-value function. And I use a parameter lambda to decide the value ratio between next move and “next next” move.

Two step lambda formula: (Set lambda=0.01)

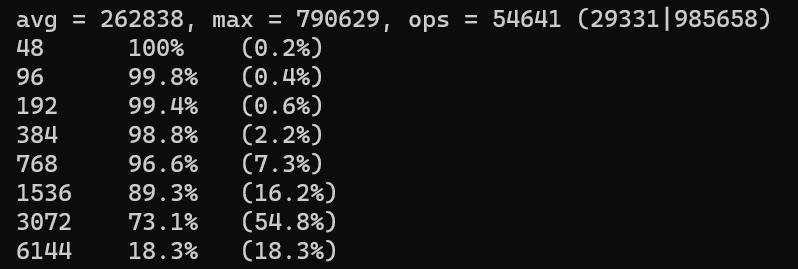
TDerr = (1-lambda)\*TDerr + lambda \* (1-lambda) \* ( reward + reward2 + get\_value(nextnext) - get\_value(next) );

TDerr \*= alpha;

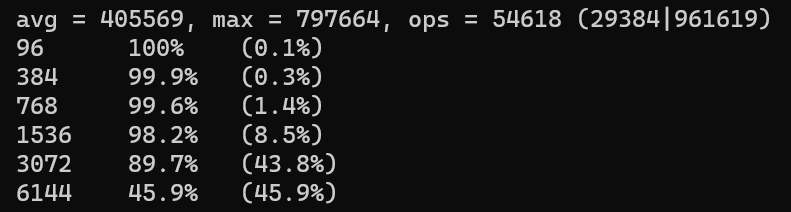
(The reason I didn’t use general TD-lambda method is that I use forward-propagation to calculate TD value, and I have no idea how to consider all the moves after current move till the end without changing my propagation model. So as simply as I can imagine, I only consider the influence of next move and the move after next move in my “Two-step TD model”.)

1. Training process

Use alpha = 0.1/32 to train for 130,000 games and results in:



Then I set alpha=0.0005 and train for another 30000 games that results in:



Finally set alpha=0.00005 and train for another 100,000 games.

