Tyler Lyczak

March 6, 2019

tlyczak2@uic.edu

CS 342

AI Algorithms

There are three main functions for determining how my AI makes a bid and move. The first function they all rely on it updateCardHandWeights(). Then the bid and play card functions called determinebid() and playCard(), respectively, use these weights generated from updateCardHandWeights() to make decisions for the AI.

The first function I want to discuss is updateCardHandWeights() because this function allows the others to function properly. In updateCardHandWeights(), it goes through each card in the AIs hand and adds each card rank (as in if it was Ace, then its rank would be 14 and if it was a 10 card, then its rank would be 10) to its suit weight. These suit weights are all private members of each AI. All the suit weights are then added to a total weight which is used to reset the value of each suit weight to a percentage out of 100. I use these percentages in later functions to determine what card to play or what bid to make.

The second function is determineBid(). This function calls updateCardHandWeights() to get the right weights for the current hand of the AI. Right away the function will return a bid of smudge if one of the weights is at 1, which means all the cards in there hand is one suit. Usually, this won’t happen so it is very uncommon. Next, the weights are added to an array list and then a for-loop will go through all the weights in the array list. In the for-loop, it contains a bunch of if-else statements to add bids to another array list depending on each percentage. The higher the percentage, the higher the bid will be. Then it will run through the bid array list and pull the highest bid out of it and return it, making that AIs bid final.

The last function playCard() calls the same updateCardHandWeights() function to determine the weights. In playCard(), it has 4 major cases that the AI will fall under depending on when the AI gets its turn. Case 1 is the AI is first to go in a new round, Case 2 is the AI is first to go after winning a trick but the round is still going, Case 3 is the AI is going when a trick is already started, and Case 4 is a fail-safe to make sure the bot will play a card if somehow all other cases fail, which they should never. Each case, besides 4, will use the weights of the AI's hand to see what cards it should play. It will also adapt to the game, like if a trump card is already played but its highest weight is in a different suit. It will use those cards of its highest suit to win tricks with no trumps or use them as throw away cards for them to pass a turn. For example, Case 3 is broken up into 6 sub-cases, such as if a trump is played but the bots trump cards are not bigger than it, it will play a lower weight suit to pass the turn to save its trump cards to win the next trick. This function analyzes the trick deck during each of its turn to determine what cards to play and will adapt to the game to win it.

In conclusion, the AI relies heavily on the updateCardHandWeights() and its ability to analyze the trick deck. It will make the smartest decision it can at the time so it can preserve its best cards and win tricks.