Theory of Computer Games, 2019 Fall Final Project

b05901020 郭彥廷

1. Implementation

1.1 Execution

```
$ make agent                    #compile the files, generate executable exec.agent
```

1.2 Implementation Structure

1.2.1 Expectiminimax Game Tree

Because flipping actions are nondeterministic, I used expectiminimax tree with Negascout search to implement the AI. If there was a covered chess, it could be all chesses that have not been flipped. Thus, I search all possible outcome and return the expected value.

For example, there are four face-down pieces at squares a6, b5, b2 and d2 in Fig. 1, and thus four chance nodes in the game tree are generated, as shown as the black circles: a6(?), b5(?), b2(?), and d2(?). The child nodes of each chance node represent the possible outcomes of a flipping action, and the branches leading from each chance node are labeled with their probabilities. Assume four pieces P, k, g ang g are in unrevealed states, so the probabilities of a6(P), a6(k), a6(g) are 1/4, 1/4 and 1/2, respectively.

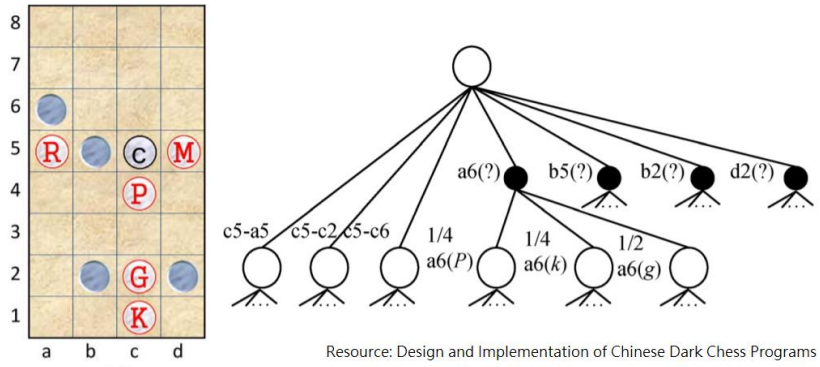

Fig. 1

Because we need to know the exact value of the child nodes to evaluate the expected value, the search window of each child node of chance nodes is set to be -INF and INF.

1.2.2 Flipping Heuristic

The problem of exceptiminimax game tree is it takes too long to search the chance nodes. Thus, I limited the search such that only one chance nodes are being searched among all child chance nodes.

To determine what nodes to be searched (which piece to flip), I evaluate all face-down pieces by their neighbor and one-step neighbor. Using TABLE II and TABLE III to give the weight to all pieces facing down and pick the highest one to flip.

TABLE II
THE WEIGHTS OF FACE-DOWN PIECES p INFLUENCED BY THEIR NEIGHBORING FACE-UP PIECE p'

p'	Weight of p	
	p' is the opponent's piece	p' is the player's piece
king	−5000	−5000
guard	−800	+800
minister	−600	+600
rook	−200	+200
knight	+100	+100
cannon	5000	−5000
pawn	−1000	−1000

TABLE III
THE WEIGHTS OF FACE-DOWN PIECES p INFLUENCED BY THEIR NEIGHBORING ONE-STEP-JUMP FACE-UP PIECE $'$

p'	Weight of p	
	p' is the opponent's piece	p' is the player's piece
king	−1000	−1000
guard	−800	−1000
minister	**−600**	−1000
rook	−400	−100
knight	−200	−100
cannon	−5000	+500
pawn	−100	−100

Source: Design and Implementation of Chinese Dark Chess Programs

For instance, the weight of the face-down piece at square b5 is -2000(= -400 – 1000 – 600) for Black's turn.

1.3 Evaluation Function

The evaluation for board B is calculated by the following formula:

$$f(B) = \begin{cases} -\boldsymbol{Inf} + 1 + \boldsymbol{Dep}, & \text{if B has no possible move} \\ c_{Basic} \times \boldsymbol{Bsc}(B) + c_{Attack} \times \boldsymbol{Atk}(B) + \boldsymbol{Rnd}, & \text{otherwise} \end{cases}$$

Where **Inf** is the width of search window, **Dep** is the tree depth from the root, **Bsc**(B) is the basic value of B, **Atk**(B) is the attack value and **Rnd** is a random number from 1 to 10.

The reason to add the depth is to choose the shortest winning path in the end games. And the reason to add a random number is to make decision among nodes that have a same value. The following subsections will discuss the basic value and attack value in my implementation.

1.3.1 Basic Value

The basic value of a board is decided by the pieces left on the board. I value the piece facing up and down the same because if facing-up pieces have more score than facing-down ones, it will flip more often and make the board complicated. The score of the related pieces, as shown in TABLE I.

The value is simply the sum of score of all the pieces.

TABLE I

king	guard	minister	rook	knight	cannon	pawn
50	30	20	10	5	25	2

1.3.2 Attack Value

To deal with end games, I noticed that the perfect distance to catch a piece is not 1 but 2, otherwise it would easily go to a draw game because the position repetition draw rule.

Therefore, to avoid playing inefficient moves, I used the influence values as shown in TABLE IV, the left-most column is the attacker piece and the first row is the defender's piece. Let X be the attacker's piece, Y be the defender piece, and MD be the Manhattan distance between X and Y. The weight value for empty square with respect to the pair (X, Y) is calculated by:

$$W(X,Y,MD) = \begin{cases} I(X,Y)/2, & if\ MD\ =\ 1 \\ I(X,Y) \times 2^{2-MD}, & otherwise \end{cases}$$

The value is added into the evaluation of the game tree. For example, the influence value for (G, m) in the left side of Fig. 3 are the numbers in the right side of Fig. 3.

TABLE IV
THE INFLUENCE-VALUES OF ATTACKER'S PIECES VERSUS DEFENDER'S PIECES

	k	g	m	r	n	c	p
K	120	108	60	36	24	48	0
G	0	106	60	36	24	48	12
M	0	0	58	36	24	48	12
R	0	0	0	34	24	48	12
N	0	0	0	0	22	48	12
C	0	0	0	0	0	0	0
P	108	0	0	0	0	0	10

Source: Design and Implementation of Chinese Dark Chess Programs

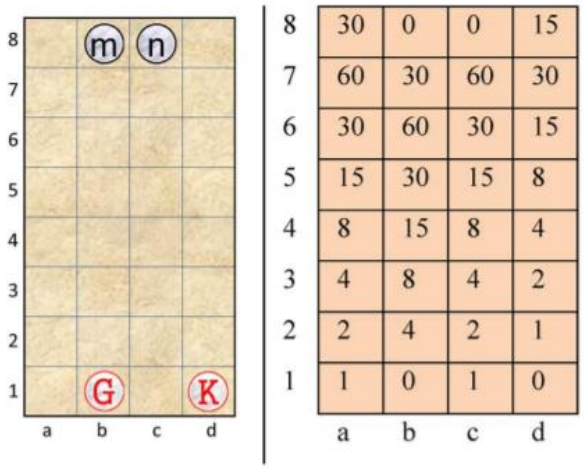

Fig. 3. An example of the influence-values.

Source: Design and Implementation of Chinese Dark Chess Programs

2. Experiments

2.1 Coefficient of Basic Value and Attack Value

I made a baseline program without applying attack value, and let 2 versions of program fight against it. The result is in TABLE V, which suggested that $c_{Basic}/c_{Attack} = 100$ is better.

TABLE V

c_{Basic}/c_{Attack}	Win	Lose	Draw
100	**70%**	**0%**	**30%**
10	60%	0%	40%

2.2 13th NTU CSIE CUP of Computer Chinese Dark Chess competition

I won 2nd place among 20 contestants, and the results are in TABLE VI. Because I had fought against rank 1 to rank 6 opponents (except rank 2 or course), my tie-breaking score is the second highest, only lower than rank 6 players, who had fought against rank 1 to rank 5 players.

In the end of the tournament, I also played against TA's champion AI Yahari in a demo game.

TABLE VI

#Games (opponent's final rank)	1st game	2nd game
1(12)	D	W
2(6)	W	L
3(10)	W	D
4(13)	W	W
5(1)	L	L
6(4)	W	D
7(5)	D	W
8(3)	D	D
9(14)	W	W
Demo(Yahari)	L	D

3. Future Work

There are still some improvements that can speed up the program, such as move ordering and end game database. Also I want to decrease the number of draw games, it should be able to avoid draw game when the rating is high in search tree.

4. Reference

[1] YEN et al.: Design and Implementation of Chinese Dark Chess Programs