BLG 435E Homework 2

1) Constraint Satisfaction Problems

Variables = Nodes = {1, 2, 3, 4, 5}

Domains = Paths = {[1-2], [2-1], [1-3], [3-1], [1-4], [4-1], [2-3], [3-2], [2-4], [4-2], [3-4], [4-3], [3-5], [5-3], [4-5], [5-4]}

Constrains = 1- Every paths must be used at most one time

2- One of symmetric paths must be used. Cannot be used both.

3- Consecutively used paths must be connected with a node. For example; after [1-2], [2-4] can be used, but paths such as [4-2] or [3-5] cannot be used.

Note: symmetric paths are paths that exits on the same edge.

2) FOL Representation

- ∀x dog (x) => animal(x)
- ∃x robot ∧ ¬carry(x, object)
- ∀x student ∧ graduated (x, high school) => graduated (x, primary school)
- ∃x students ∧ ¬(x, AI course)
- ∃x∀y table(x) ∧ exist(x), table(y) ∧ exist(y) => x=y
- ∃x∀y teacher(x) ∧ teacher(y) ∧ talks (x, y) => teaches(y, physics) ∧ x≠y

FOL and Resolution

3) In the minimax solution of the problem, moves are performed sequentially with the dfs method. When there is no more moves to be made, if the last move done by 1st player, we calculate the maximum, else we calculate the minimum. For this problem if 1st player wins minimax of that node is 1. Else if the 2nd player wins minimax of that node is 0. After reaching the last child, the minimax value of the previous node is calculated. While calculating, the minimum or maximum is calculated according to the player who making the move. This way all nodes are checked and the maximum of the root is found. In this way, when the best moves are made, it is determined who will win the game. For this problem, I keep as the minimax value 1 when player 1 wins, and minimax value 0 when player 2 wins.

When we add alpha-beta pruning, unnecessary branches are trimmed. That is, when a minimum computed node has a minimum (0 in this problem), or a max computed node has a max (1 in this problem), other children will not controlled. In this way, the same result is obtained faster by saving time.

For the given input file there are 12 empty square and minimax algorithm didn't end (so long). I tested both algorithms for 9 and 10 empty square input.

For 9 empty square:

Number of nodes generated in minimax : 406242

Runtime of minimax: approximately 5 sec.

Number of nodes generated in minimax with alpha-beta : 1569

Runtime of alpha-beta: less than a sec.

For 10 empty square:

Number of nodes generated in minimax : 6582314

Runtime of minimax: approximately 2 min.

Number of nodes generated in minimax with alpha-beta : 17019

Runtime of alpha-beta: less than a sec.

Compiling the code: `g++ minimax.cpp –o out`

Running the program:

For alpha-beta the program takes two arguments: first is name of the input file, second is alpha.

For basic minimax the program takes one argument: name of the input file

Talha Çomak

150160726