Problem Formulation

3 different Ways to solve Tic-Tac-Toe Game in Artificial Intelligence

Problem Definition:

What are the 3 different ways of solving the Tic-Tac-Toe Problem applying AI? Show the improvements obtained from one over the other using better knowledge representation.

Solution:

The tic-Tac-Toe game can be solved in three ways. The programs or solutions in the Series increase in,

- Their complexity
- Use of generalization
- Clarity of their knowledge
- Extensibility of their approach

Program or Solution 1 to Tic-Tac-Toe Game:

Following are the data structures used in the program or solution:

Board: 9 element vectors representing the board, with 1-9 for each square. An element contains the value 0 if it is blank, 1 if it is filled by X, or 2 if it is filled with an O

A large vector of 19,683 elements (3^9), each element is a 9-element vector.

Algorithm:

1. View the 9 element vector as a ternary number. Convert ternary number to a decimal number.
2. Use the computed decimal number as an index in the Move-Table and access the vector stored there.
3. Set the new board to that vector.

Comments on program or solution 1:

Advantage: This program is very efficient in time.

Disadvantages:

1. A lot of space is required to store the Move-Table.
2. A lot of work is needed to specify all the entries in the Move-Table.
3. Difficult to extend to 4×4 puzzle.

Program or Solution 2 to Tic-Tac-Toe Game:

Following are the Data Structure used in Program or solution 2:

A nine-element vector is used for representing the board. But instead of using 0,1 and 2 in each element, we store 2 for blank, 3 for X, and 5 for O.

Following are the list of Functions in program or solution 2:

This function returns 5 if the center square is blank. Else any other blank square.

Posswin(p): This function returns 0 if the player p cannot win on his next move; otherwise it returns the number of the square that constitutes a winning move. If the product is 18 (3x3x2), then X can win. If the product is 50 (5x5x2) then O can win.

Go(n): This function Makes a move in the square n.

Strategy used in program or solution 2:

Turn = 1 Go(1)

Turn = 2 If Board[5] is blank, Go(5), else Go(1)

Turn = 3 If Board[9] is blank, Go(9), else Go(3)

Turn = 4 If Posswin(X) ¹ 0, then Go(Posswin(X))

and so on.

Comments on program or solution 2:

1. The proposed solution is not efficient in terms of time, as it has to check several conditions before making each move.
2. Easier to understand the program’s strategy.

3. Hard to generalize the program or solution.

Program or Solution 3 to Tic-Tac-Toe Game:

In this case, the next move is selected by seeing the next possible board position that results from each possible move.

This function returns 5 if the center square is blank. Else any other blank square.

Posswin(p): This function returns 0 if the player p cannot win on his next move; otherwise it returns the number of the square that constitutes a winning move. If the product is 18 (3x3x2), then X can win. If the product is 50 (5x5x2) then O can win.

Go(n): This function Makes a move in the square n.

Strategy used in program or solution 2:

Turn = 1 Go(1)
Turn = 2 If Board[5] is blank, Go(5), else Go(1)
Turn = 3 If Board[9] is blank, Go(9), else Go(3)
Turn = 4 If Posswin(X) ¹ 0, then Go(Posswin(X))
and so on.

Comments on program or solution 2:

1. The proposed solution is not efficient in terms of time, as it has to check several conditions before making each move.
2. Easier to understand the program’s strategy.
3. Hard to generalize the program or solution.

Program or Solution 3 to Tic-Tac-Toe Game:

In this case, the next move is selected by seeing the next possible board position that results from each possible move.