SE 14 - High Level Description -

Nino Lindenberg

Adversarial Search and/or Game Problem

Problem Description:

In TicTacToe, there are 2 players (the AI will take either MAX or MIN) that compete against each other. MAX aims to maximize its value and MIN aims to minimize its value. The game starts with an empty board as the initialize state and the players take turn in filling the squares on the board. The game ends when it reaches the terminal state that is defined by either a “draw” when no one wins and the board is fully filled or “MAX or MIN wins” if one of them manage to put X for MAX or O for MIN in 3 consequtive times in a diagonal, vertical, or horizontal line.

Area of AI: Planning

Applied Algorithms:

- Minimax

  A recursive algorithm that aims to find the most optimal action for either MAX or MIN by iterating over the whole leaf nodes in the game search tree to evaluate all possible values. For the AI as MAX player, the value wanted is either bigger than MIN or at least equal to the value of MIN and exactly the opposite for the AI as MIN player. The time complexity for Minimax is $O(b^n)$ and the space complexity is $O(bn)$.

- Alpha-Beta Pruning

  The optimization of the “Minimax Algorithm” by adding the helper values: $\alpha = -\infty$ and $\beta = \infty$, and also the functionality to “cut-off” the rest of the leaf nodes in the whole tree as soon as, for alpha, alpha is bigger than or equal to the value of beta, or, for beta, beta is smaller than alpha or at least equal to alpha. In other words, alpha-beta attempts to calculate the most optimal action without looking at every game search tree. The time complexity for Minimax is $O(b^{n/2})$ and the space complexity is $O(bn/2)$.

Results:

- The AI is unbeatable. The minimum outcome that can be expected is a draw, or you lose. There is no way a player can win the game.

- Using minimax, the total thinking time of the AI as MAX player is around 1.28 seconds. AI as the MIN player, the total time is 10.61 seconds.
- Using alpha-beta pruning, the total thinking time of the AI as MAX player is around 0.24 seconds. AI as the MIN player, the total time is 0.874 seconds.

Location: https://github.com/Artificial-Ninoligence/SE14_AI-Basics/tree/main/01_Planning

---

Knowledge and Reasoning Problem

Problem Description:

The "Knights and Knaves" logical puzzle in which the AI needs to guess who is the knight and who is the knave based on the knowledge base given by the game rules and the sentence said by the character A, B, or C.

Area of AI: Reasoning

Applied Algorithms:

- Inference Algorithm with Model Checking Approach

  The algorithm works by enumerating all possible models to check that α is true in all models in which KB is true. Models here means the assignment of the value True and or False to every propositional symbol.

Results:

The AI can guess accurately who is the knight and who is the knave.

Location: https://github.com/Artificial-Ninoligence/SE14_AI-Basics/tree/main/02_Knowledge%26Reasoning

---

Constraint Satisfaction Problem

Problem Description:

- The problem in the crossword puzzle is the assignment of each different word for each variable such that the unary, binary, and preference constraint are satisfied:
  - Unary Constraint - Constraint by the length of the variable (number of letters), i.e. for variable 1, the value "Eyes" would satisfy the constraint, but "Eye" would.

- Binary Constraint - Constraint on variables that verlap with their neighbours, i.e. variable 1 has a single neighbour: Variable 2. Variable 2 has 2 neigbours: Variable 1 and variable 3. Each of these 3 variables will be sharing a single square that is common to them both, hence they are overlapping.
- Preference Constraint - The same word can’t be repeated in the puzzle.

- A solution for CSP is achieved through an assignment of values to the variables that satisfies all constraints.

Area of AI: Optimization

Applied Algorithms:

- Backtracking Search

  This algorithm is modeled on the recursive version of depth-first search Backtracking search only keeps a single representation of a state and alters that representation rather than creating new ones. In Backtracking search algorithm for CSP, the algorithm repeatedly chooses an unassigned variable and iterates over all possible values in the domain of that variable (in turn), trying to extend each one of them into the solution in a recursive manner. If the execution is successful, it returns the solution. If it failed, it will restore the previous state and tries the next assignment. At the end, if no value works for the assignment, then it will return “No Solution”. The time complexity is $O(2^n)$.

Results:

The AI can assign all values to the variables while satisfying the arc-constraint and the preference constraint. This means that the crossword puzzle are filled with unique words (1 words 1 time) and no conflict at the overlapping square in the puzzle. If the puzzle is modified, there are some structures where the AI either cannot assign all values to the variable, or the constraints are not met, hence it will return “No solution”.

Location: https://github.com/Artificial-Ninoligence/SE14_AI-Basics/tree/main/03_Optimization

---

Supervised Learning Problem - Image Detection

Problem Description:

Gears used in industrial machineries are often broken in a specific time period. If it is not predicted, then it can break the whole machine. With deep learning (neural network), we can predict if there is any metal fatigue on the gear to avoid the ttal break down of the machines.

Area of AI: Learning

Applied Algorithms: Multi-Class Classification Algorithm with Deep Learning

Results:

The deep learning based object detection model successfully detect all labels successfully (up to 97%):

- Gear label => any gear on the image?
- Metal fatigue label => broken, missing, scratched, broken missing, broken scratched, missing scratched

  The model is not only detecting but it also recognises the location of the objects/the metal fatigues (if any).

Location: -