PROJECT REPORT

CS 154

BACKGAMMON

SUBMITTED BY:

Ashish Sonone (110050022)

Dhanesh Kumar (110050021)

Himanshu Roy (110050019)

PROBLEM DESCRIPTION:

The problem mainly consists of developing "AI" for single player "Backgammon" . It also provides a friendly environment for playing the game .

DESIGN OF PROGRAM

The game consists of two levels :

Level 1 : In this level the AI is essentially random.

Level 2 : In this level the AI uses the evaluation-function and expecti-minimax to choose the best move from possible moves.

The evaluation-function assigns values to every board state according to future projections , strategy and prior knowledge of the gameplay.The value represents the relative advantage that the computer has over opponent .

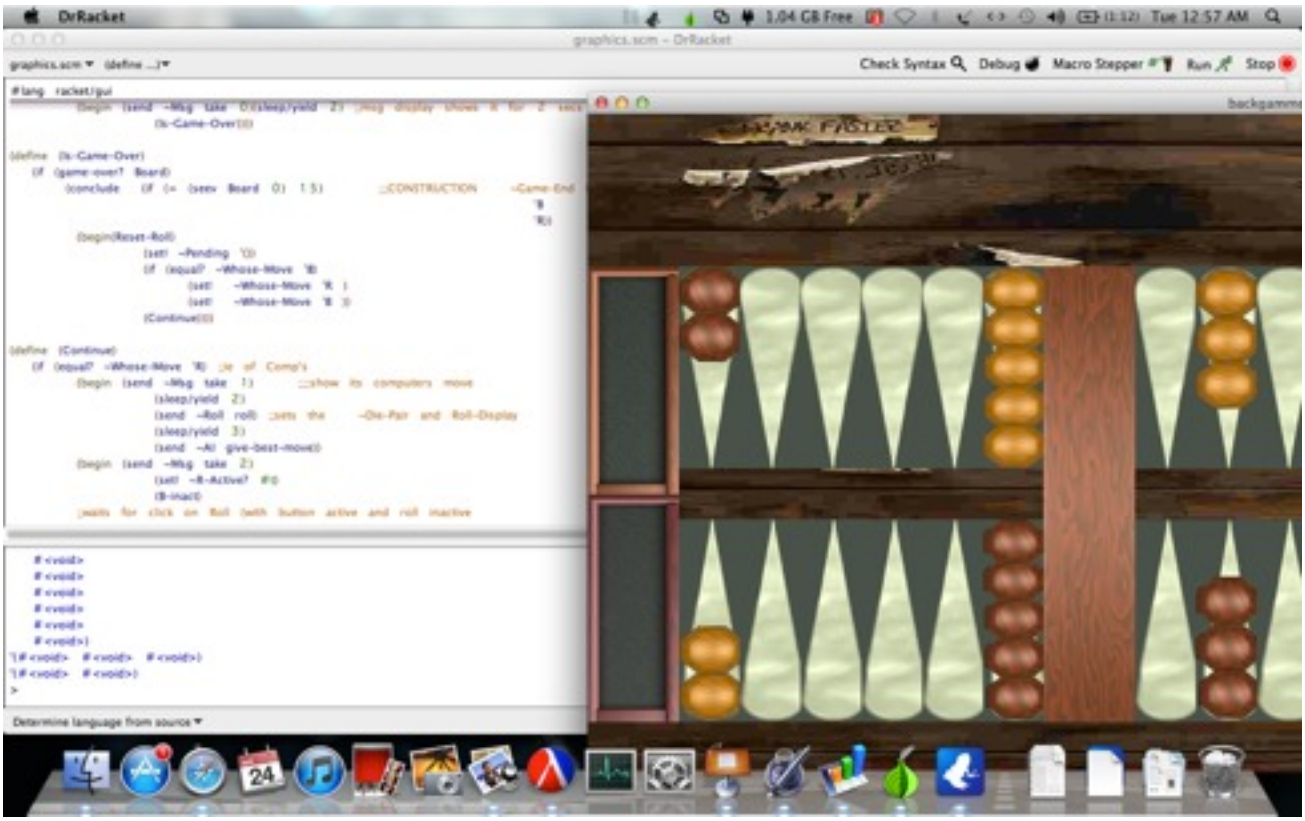
Our AI looks two moves ahead and generates a game tree of depth 2 that consists of max node, chance node , min node and finally the leaves.

The game tree is generated the following way:
Max node has current board state.(Computer to make move)
Now look at all possible moves computer can make with die-roll that has come.Each of them represent chance node
Each chance node has 21 children reprenting possible 21 dice rolls .These children are called min nodes.
For each min node (User to make move) generate possible table state corresponding to die-value associated to it.

Choosing the best move:
Basic rules of evaluation are :

At max node choose the highest valued child
At min node choose the minimum valued child and give its value
At chance node calculate the expectation of its children using the probability distribution of different dice-rolls.
At leaf : evaluate the board using evaluation function.

The AI searches the game tree and selects the move that would lead to the most advantageous state . It uses depth-first search and modified alpha beta pruning(for chance node) to minimize the search in the tree as the tree gets very huge even at depth two.So evaluating each leaf is not feasible as it make program slow.



Sample Input And Output

We have developed a user-friendly interface . So there is no need for sample input-output.

Limitations and Bugs

Our AI can't search more than two depths in the search tree.
The evaluation function is not based on regressive probabilistic evaluation and doesn't consider some special conditions. The evaluation function can be improved a with little more effort and insight.
Also the expectiminimax search method can be optimized so that it searches more depth in same time.

Any other point of interest

Many GUI classes (like canvas , button , panel , canvas) have been modified to our own purposes so that containees can be placed at specified position in panel.

The game has much scope of further development of AI . More and more probabilistic analysis improves some parts of the game like the "end game" where both players are fairly independent , evaluation function can be very effectively used to setup best possible configuration.
Also there is a scope for self learning , which can aid the probabilistic part . Since the game some uncertainty also because of the dice.