LEARNING GAMES FROM VIDEOS GUIDED BY DESCRIPTIVE COMPLEXITY

Summarized by: Akshay Agrawal (10058)

Objective: To learn board games (Double player games only; like Tic-Tac-Toe, Connect4, Pawns, Breakthrough) from a given set of videos.
Set of videos includes cases when:
Player 1 wins, Player 2 wins, illegal moves are made, won by none
(The number of training videos of each kind varies proportionately to the number of permutations of each kind possible)

Approach suggested:
(All the Descriptions made are with respect to a 3*3 grid, Descriptions may vary with the choice of board structure (e.g. triangle, grid etc.))

- **Collection of data from observation into modifiable data structures:**
  Data structure used - Relational Structure
  Description of 'Relational Predicates' used:
    R(x,y) => (&y == &x+3) : 'TRUE' if y follows x along a row, 'FALSE' otherwise
    C(x,y) => (&y == &x+1) : 'TRUE' if y lies above x in a column, 'FALSE' otherwise {This definition is only descriptive, lacks an additional factor}
    diag(x,y) => 'TRUE' if for some 'u' (R(x,u) && C(u,y)), 'FALSE' otherwise {i.e. 'u' lies right of 'x' and 'y' above 'u'}
    lastRow(x) => 'TRUE' if for no 'u' C(x,u), 'FALSE' otherwise {i.e. no 'u' exists below 'x'}
  All these predicates and may be some more, corresponding to a certain state of a game are filled up by image analysis results

- **Designing appropriate logic formulas ("RULES") as per the conditions for win or loss or tie learnt:**
  Considering each video as a sequence of structures, all possible pairs are classified under appropriate condition, depending on the label of the video being learnt (1Win, 2Win, Illegal, No Win etc.)
  - Learning Winning Condition:
    e.g. For some x1(W(x1)^For no x0:C(x1, x0)) implies "White wins if there is no other piece below x0 for atleast one position of white piece"
  - Learning Legal Moves/ Illegal moves

- **Game Playing Algorithm:**
  Algorithm to decide the next move based on the expectation of the player by analysing the possible states after each move and selecting the most adequate ones from the set of interest.
  Technically this is done by assuming the current structure/universe as a node of UCT tree, and all the possible legal moves from this state as the children nodes to this node.
  Then probabilities of win|loss|tie are determined along each path of the tree and the one with most P(win) is adopted, and accordingly a move is made.
Drawback: Can prove to be very slow for a larger game structure with more relational structures defining the states.

Major Contribution:
- Following equivalences are used: Games as mathematical structures and moves as logical rules.
- Resizable number of 'Relational Predicates' can be used.
- Logic is used for rules definition (Second-order logic with additional real arithmetic has been used).

References:
- http://toss.sourceforge.net/eval.html
- http://toss.sourceforge.net/create.html