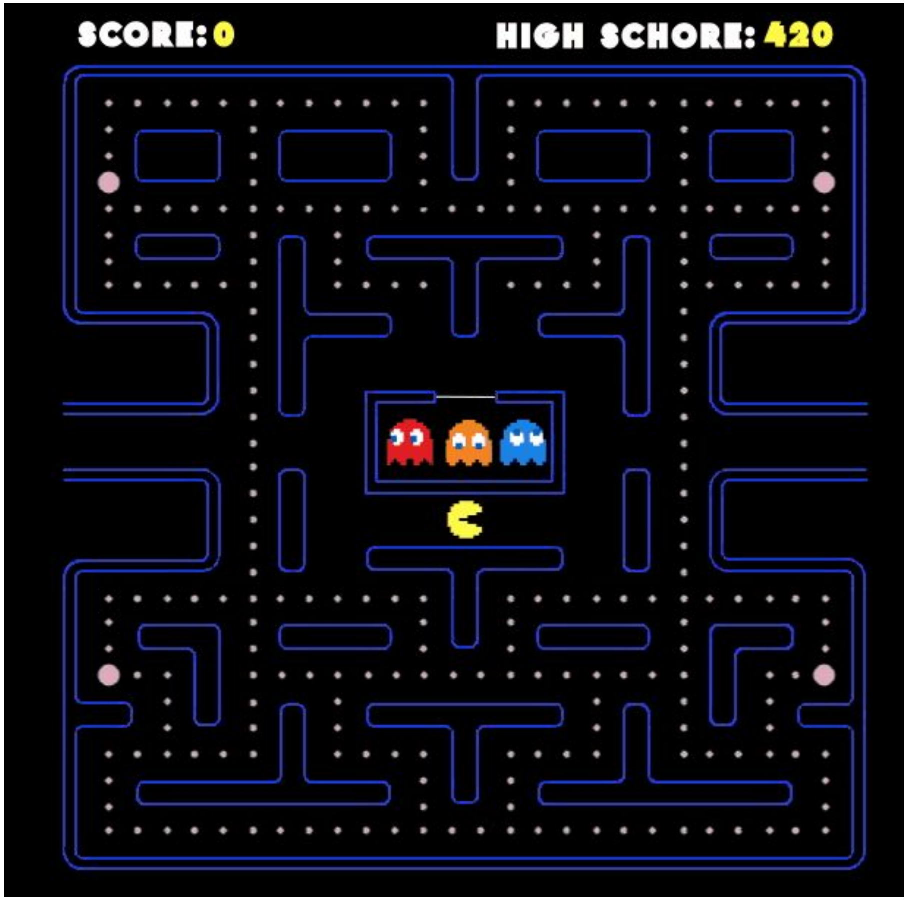
Fast Approximate Max-n Monte Carlo Tree Search for Ms Pac-Man

Spyridon Samothrakis, David Robles, and Simon Lucas, Senior Member, IEEE

Ms Pac-Man

- Released in 1981, it became an immensely popular predator/prey like game due to introduction of element of randomness to ghosts
- It requires short term planning and reactive skill
- It provides a platform that is both simple enough for research and complex enough to require intelligent strategies for gameplay

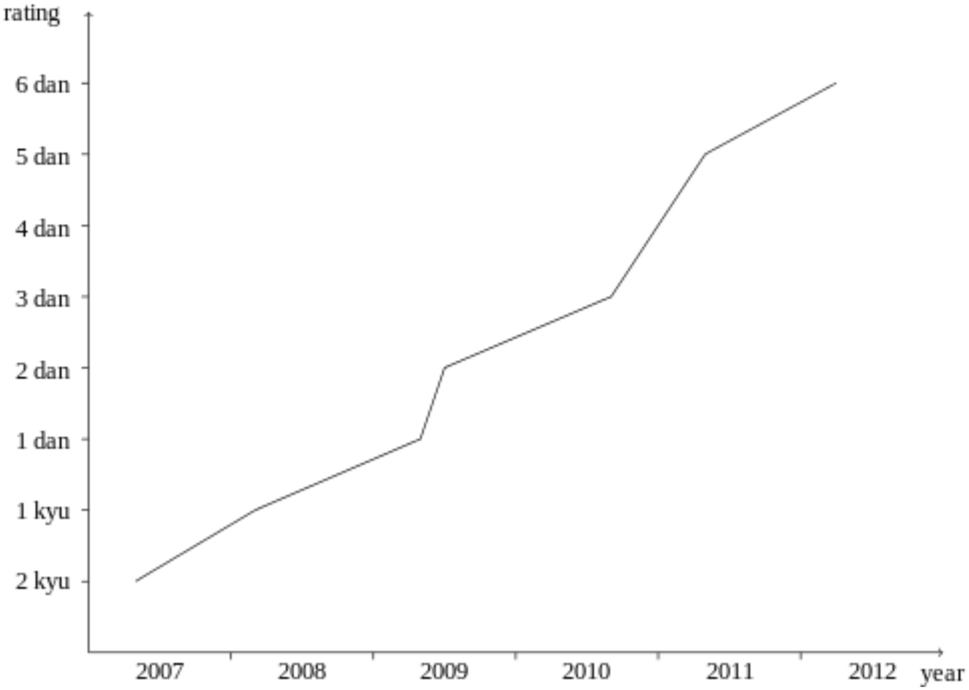


Previous Research

- Bonet and Stauffer model
    - Neural networks and temporal difference learning on a simple grid
    - Basic ghost avoidance
- Gallagher and Ryan model
    - Simple FSM model with set of rules to control movement
    - Weight parameters in the rules evolved using PBIL algorithm
    - Achieved machine learning at a minimum level
- Robbles and Lucas model
    - First attempt to apply tree search
    - It expanded a route-tree based on possible moves that the agent can take, upto a depth of 40
    - Hand-coded heuristics were used to evaluate the paths
    - High score of 40000

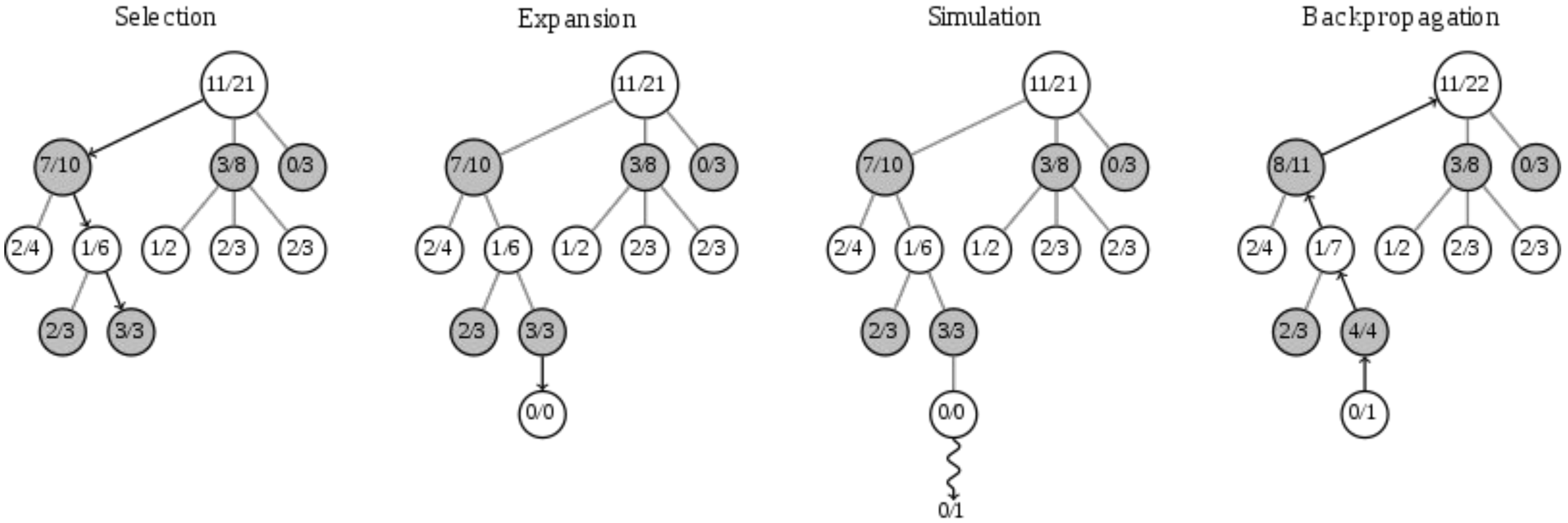
History

- The Monte Carlo method, which uses randomness for deterministic problems difficult or impossible to solve using other approaches, dates back to the 1940s
- Bruce Abramson explored the MCTS idea in his 1987 PhD thesis and said it "is shown to be precise, accurate, easily estimable, efficiently calculable, and domain-independent.”
- In March 2016, AlphaGo was awarded an honorary 9-dan (master) level in 19×19 Go for defeating Lee Sedol in a five-game match with a final score of 4-1



Monte Carlo Tree Search

- Approximation of future rewards sense can be achieved through random sampling
- The agent extrapolates to future states in a random fashion and moves to the state with the highests predicted reward
- Stochastic form of best first search



Exploration vs Exploitation

- Maintain balance in the selection of nodes with high win-rate and nodes with few simulations
- UCT (Upper Confidence Bound in trees) is the first formula introduced for balancing exploration and exploitation in games
- At each node of the game tree, the move for which the expression

  $w_i/n_i + c\sqrt{(\ln(N_i)/n_i)}$ has the highest value is chosen.

  w_i - no. of wins for the node considered after the ith move

  n_i - no. of simulations for the node considered after the ith move

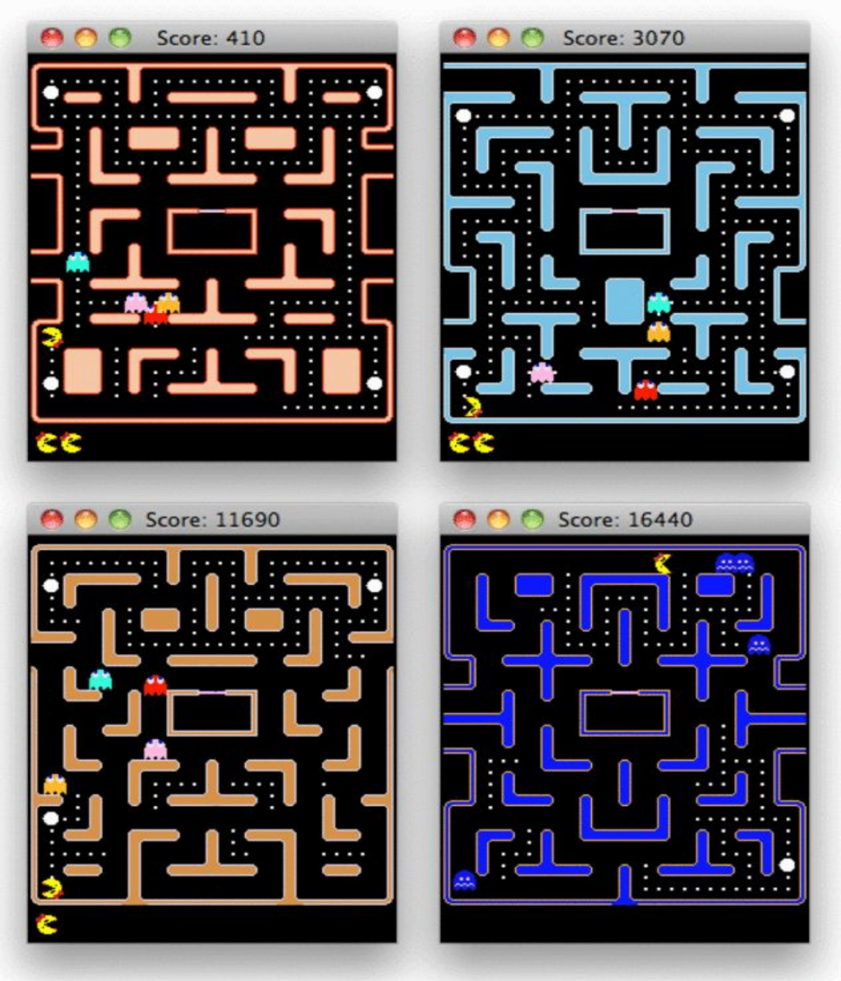
  N_i - total no. of simulations after the ith move

Applying MCTS in Pac-Man

- MCTS can be applied only on turn based games
- We model Pac-Man as a five-player game, and base the tree on max-n
- Pac-Man is a simultaneous move game, at least theoretically speaking, none of the min-max like trees is really applicable
- In order to solve the problem of not having a natural end state, we artificially limit the search tree to a fixed depth. An end node can be either the natural end of the game (a ghost eats pac-man) or the end of a tree, with tree_depth = c
- Each player tries to maximize its payoffs independently from the rest
- A simple efficient algorithm is run to compute the shortest-path distance between every node and every other node in the mazes. These distances are stored in a lookup-table, and allow fast computation of the various controller-algorithm input features

Gameplay

- The movement of the ghosts and the Pac-man agent are controlled by respective controller-algorithms
- The mazes of the game are modelled as graphs of connected nodes
- Each node has two, three or four neighboring nodes
- Each maze is played twice consecutively, starting in Maze A continuing through to Maze D
- When Maze D is cleared, the game goes back to Maze A and continues the same sequence, i.e., (A,A,B,B,C,C,D,D,A,A,...) until game over



Advantages and Disadvantages of MCTS

- It does not require an explicit evaluation function
- It can be employed in games without a developed theory
- It achieves better results than classical algorithms in games with a high branching factor
- It may not see a single branch that leads to a loss as it may be difficult to find it at random. The search may not take it into account and hence lead to a loss. (AlphaGo's loss in its fourth game against Lee Sedol)

Conclusion

- MCTS can successfully be used in a real time game, getting results that are almost two orders of magnitude better than previous results in the same simulator acquired by evolutionary, reinforcement learning and genetic programming methods
- The plan the agent is to follow is re-formed at every timestep, making the need for feedback corrections redundant
- More computational time, which invariably results in more simulations, does not necessarily result in better performance
- The approach presented in this paper has great potential for creating generic AI agents. One can easily envisage a procedure where the most important abstract features of a world are modelled and given to an agent to reason with

Thank you!

Q&A