General Game Playing
Statistical Search

Michael Genesereth
Computer Science Department
Stanford University
Evaluation functions examined thus far
Based on properties of small portions of game tree
e.g. mobility, intermediate state values

Statistical Methods
Based on statistical analysis
Based on larger portions of game trees

Methods:
Monte Carlo Search
Monte Carlo Tree Search (in particular UCT)
Monte Carlo Search
(1) Optionally explore game graph to some level.

(2) Beyond this, explore to end of game from fringe nodes, making random choices for moves of all players.

(3) Assign expected utilities to fringe states by summing utilities and dividing by number of trials.
Example
Example
function maxscore (state, level)
    if (findterminalp(state, ruleset))
        return findreward(role, state, ruleset);
    if (level>levels) {return montecarlo(state)};
    var actions = findlegals(role, state, ruleset);
    var score = 0;
    for (var i=0; i<actions.length; i++)
        {var result = minscore(actions[i], state, level);
            if (result==100) {return 100};
            if (result>score) {score = result}
        }
    return score}
function montecarlo (state)
    {var total = 0;
        for (var i=0; i<count; i++)
            {total = total + depthcharge(state)};
        return total/count}

function depthcharge (state)
    {if (findterminalp(state, ruleset))
        {return findreward(role, state, ruleset)};
        var move = seq();
        for (var i=0; i<roles.length; i++)
            {var options = findlegals(roles[i], state, library);
                var best = randomindex(options.length);
                move[i] = options[best];
            }
        var newstate = findnexts(move, state, library);
        return depthcharge(newstate)}
Problems

  Optimistic - opponent might not respect probabilities
Problems

- Optimistic - opponent might not respect probabilities
- Does not utilize game structure in any useful way
Problems and Features

Problems
   Optimistic - opponent might not respect probabilities
   Does not utilize game structure in any useful way

Benefits
   Fast because no branching
   Small space because nothing stored in probes
   Provides guidance when other heuristics fail
Monte Carlo Tree Search
Monte Carlo Tree Search (MCTS) is a search method that relies on random probes to estimate state values. Blend of Greedy and MCS.

Like MCS:
- Builds up game tree incrementally
  - Random probes to end of game to estimate state values

Like Greedy:
- MCS expands tree uniformly
- MCTS expands nonuniformly like Greedy
Overview

Select
Overview

Select → Expand
Overview

Select → Expand → Simulate → Backpropagate
Overview

Select → Expand → Simulate → Backpropagate

Repeat until timeout
Single Player
Select
Select

Diagram:

- Root: (u=180, v=7)
  - u=100, v=3
    - u=0, v=1
    - u=100, v=2
  - u=80, v=3
    - u=0, v=1
  - u=0, v=1
  - u=50, v=1
  - u=30, v=1
Select

\[ \text{selectValue} = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}} \]
According to the diagram and the equation:

\[
\text{selectValue} = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}}
\]

The process is referred to as **Exploitation**.
Select

\[ \text{selectValue} = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}} \]

Exploitation  Exploration
Select

\[
\text{selectValue} = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}}
\]

Exploitation  Exploration
Select

\[
\text{selectValue} = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}}
\]

Exploitation  Exploration
Select

\[
\text{selectValue} = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}}
\]

Exploitation  Exploration
Select
Simulate
Backpropagate
Backpropagate
Backpropagate
Backpropagate
Backpropagate
Multiple Players
Multiple Players
Multiple Players

\[ \text{selectValue} = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}} \]
Multiple Players

\[
\text{selectValue} = \frac{-1 \times \text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}}
\]
Multiple Players

```
- u = 160
  - v = 4
    - u = 100
      - v = 2
    - u = 80
      - v = 1
- u = 60
  - v = 2
    - u = 10
      - v = 1
    - u = 50
      - v = 1
- u = 20
  - v = 1
```
Multiple Players

- \( u = 160 \), \( v = 4 \)
- \( u = 60 \), \( v = 2 \)
- \( u = 10 \), \( v = 1 \)
- \( u = 50 \), \( v = 1 \)
- \( u = 100 \), \( v = 2 \)
- \( u = 80 \), \( v = 1 \)
- \( u = 20 \), \( v = 1 \)
Multiple Players

$u = 160$
$v = 4$

$u = 60$
$v = 2$

$u = 10$
$v = 1$

$u = 100$
$v = 2$

$u = 80$
$v = 1$

$u = 50$
$v = 1$

$u = 20$
$v = 1$

selectValue = \frac{\text{node.utility}}{\text{node.visits}} + C \times \sqrt{\frac{\ln(\text{node.parent.visits})}{\text{node.visits}}}$
Multiple Players
Why is MCTS better than MCS?
Why is MCTS better than MCS?

<table>
<thead>
<tr>
<th>Move</th>
<th>Depth Charges</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0, 0, 0, 0, 0, 0</td>
</tr>
<tr>
<td>B</td>
<td>60, 60, 60, 60, 60</td>
</tr>
<tr>
<td>C</td>
<td>0, 100, 0, 100, 0, 100</td>
</tr>
</tbody>
</table>
Why is MCTS better than MCS?

<table>
<thead>
<tr>
<th>Move</th>
<th>Depth Charges</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0, 0, 0, 0, 0, 0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>60, 60, 60, 60, 60, 60</td>
<td>60</td>
</tr>
<tr>
<td>C</td>
<td>0, 100, 0, 100, 0, 100</td>
<td>50</td>
</tr>
</tbody>
</table>
Why is MCTS better than MCS?

<table>
<thead>
<tr>
<th>Move</th>
<th>Depth Charges</th>
<th>Mean</th>
<th>Std dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0, 0, 0, 0, 0, 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>60, 60, 60, 60, 60, 60</td>
<td>60</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0, 100, 0, 100, 0, 100</td>
<td>50</td>
<td>54.8</td>
</tr>
</tbody>
</table>
https://en.wikipedia.org/wiki/Monte_Carlo_tree_search