Outline

- Games of chance
- Perfect play
- Resource limits
- Alpha-beta pruning
- $\alpha - \beta$ pruning
Pruning to reduce costs (McCarthy, 1956)

Samuel, 1955-57)

Finite horizon, approximate evaluation (Zuse, 1945; Shannon, 1950)

Algorithm for perfect play (von Neumann, 1944)

Plan of attack:

Time limits unlikely to find goal, must approximate

"Unpredictable" opponent solution is a contingency plan

Games vs. Search Problems
<table>
<thead>
<tr>
<th>Deterministic</th>
<th>Imperfect Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>nuclear war</td>
<td>bridge, poker, scrabble</td>
</tr>
<tr>
<td>monopoly</td>
<td>go, othello</td>
</tr>
<tr>
<td>backgammon</td>
<td>chess, checkers,</td>
</tr>
<tr>
<td>chance</td>
<td></td>
</tr>
</tbody>
</table>

**Types of Games**
E.G., 2-Play Game:

- Minimax

Perfect play for deterministic, perfect-information Games
function \textsc{Minimax-Value}(state, game) returns a utility value

\begin{align*}
\text{return \ \text{the } \text{lowest } \text{Minimax-Value of } \text{Successors} \ (\text{state})} \\
\text{else} \\
\text{return \ \text{the highest } \text{Minimax-Value of } \text{Successors} \ (\text{state})} \\
\text{else if } \text{Max is to move in } \text{state} \text{ then} \\
\text{return} \ \text{Utilty} \ [\text{game} \ (\text{state})] \\
\text{if Terminal-Test} \ [\text{game} \ (\text{state})] \ then \\
\text{return a utility value} \\
\text{end}
\end{align*}

\begin{align*}
\text{for each } \text{op in } \text{Operators} \ [\text{game}] \ do \\
\text{Minimax-Value} \ [\text{op} \rightarrow \text{Minimax-Value} \ [\text{append} \ (\text{op}, \ \text{game})], \ \text{game})] \\
\text{end}
\end{align*}

\text{function} \ \text{Minimax-Decision} \ [\text{game}] \ \text{returns an operator}
Properties of minimax

For chess, \( q \approx 37 \), in \( 100 \) for "reasonable" games

\[
\text{Space complexity} \approx O(n^m) \quad \text{(depth-first exploration)}
\]

\[
\text{Time complexity} \approx O(n^m)
\]

Optimal? Yes, against an optimal opponent. Otherwise?

Complete? Yes, if tree is finite (chess has specific rules for this)

For exact solution completely infeasible
Resource Limits

Standard approach:

\[
\begin{align*}
\text{Suppose we have 100 seconds, explore 10^4 \text{ nodes/second}}
\end{align*}
\]

\[
\begin{align*}
10^6 \text{ nodes per move } \iff
\end{align*}
\]

\[
\begin{align*}
\text{e.g., depth limit (perhaps add quiescence search)}
\end{align*}
\]

\[
\begin{align*}
\text{evaluation function}
\end{align*}
\]

\[
\begin{align*}
\text{= estimated desirability of position}
\end{align*}
\]
\[
\text{etc.}
\]
\[
(s)^1 f^1 m = (s)^1 f^1 m
\]
\[
(s)^u f^u m + \cdots + (s)^z f^z m + (s)^1 f^1 m = (s)^{\text{EVAL}}
\]

For chess, typically linear weighted sum of features:

Black winning

White to move

White slightly better

Black to move
Payoff in deterministic games acts as an ordinal utility function.

Only the order matters:

Behaviour is preserved under any monotonic transformation of \( \text{EVAL} \).

\[
\begin{align*}
\text{MIN} & : 400 \\
\text{MAX} & : 20
\end{align*}
\]
12-ply ≈ Deep Blue, Kasparov
8-ply ≈ Typhoo PC, human master
4-ply ≈ human novice

4-ply lookahead is hopeless chess player!

\[ a = \alpha \left( \beta = b \right) \]

Does it work in practice?

2. \textsc{Utility} is replaced by \textsc{Eval}
1. \textsc{Terminal} is replaced by \textsc{_cutoff}
\textsc{Minimax cutoff} is identical to \textsc{MinimaxValue} except

Cutting off search
Pruning example
\[
\begin{align*}
\text{MAX} & \quad 14 & x & x & 2 & 8 & 12 & 3 & \text{MIN} \\
\text{MIN} & \quad 14 & 2 & 3 & \text{MAX}
\end{align*}
\]
are relevant (a form of metareasoning)

A simple example of the value of reasoning about which computations

can easily reach depth 8 and play good chess

doubles depth of search

\( O(\sqrt{n/m}) \)

With "perfect ordering," time complexity improves effectiveness of pruning

Good move ordering improves effectiveness of pruning

Pruning does not affect final result

Properties of \( \delta \)
Define $g$ similarly for MIN.

If $v$ is worse than $v'$, MAX will avoid it, prune that branch.

$v$ is the best value (to MAX) found so far off the current path.

$\text{WHY is it called } v'$?
The \( u = \beta \) algorithm

 Basically MINIMAX + keep track of \( u, \beta + \) prune
to suggest plausible moves.

Go: human champions refuse to compete against computers, who are too good.

Othello: human champions refuse to compete against computers, who extending some lines of search up to 40 ply.


Checkers: Chinook ended 40-year reign of human world champion Marvin Thompson by winning 443,748,401,247 positions.

Deterministic games in practice
Non-deterministic Games

Simplified example with coin-flipping instead of dice-rolling:
E.g., in backgammon, the dice rolls determine the legal moves.

MIN

MAX

CHANCE

-2
0.5
-1
0.5
0
0.5
1
0
5
6
4
7
2
2
0
A version of α–β pruning is possible

... return average of EXPECTEDMINMAX - VALUE OF SUCCESSORS(state)

If state is a chance node then

... just like MINMAX, except we must also handle chance nodes

EXPECTDMINMAX gives perfect play

Algorithm for nondeterministic games
A world-champion level
TDCAMON uses depth-2 search + very good EVAL

α−β pruning is much less effective

As depth increases, probability of reaching a given node shrinks

\[ \text{depth} = 20 \times 1.2 \times 10^9 \]

Backgammon \(\approx 20\) legal moves (can be 6,000 with 1-I roll)

Dice rolls increase with 2 dice

Non-deterministic games in practice
Hence $EVAT$ should be proportional to the expected payoff.

Behaviour is preserved only by positive linear transformation of $EVAT$.
Games are to AI as grand prix racing is to automobile design.

Uncertainty constraints the assignment of values to states.

Good idea to think about what to think about.

Perception is unattainable; must approximate.

They illustrate several important points about AI.

Games are fun to work on! (and dangerous)

Summary