Adversarial Search

George Konidaris
gdk@cs.brown.edu
“Chess is the Drosophila of Artificial Intelligence”
Kronrod, c. 1966

TuroChamp, 1948
Games


“The chess machine is an ideal one to start with, since: (1) the problem is sharply defined both in allowed operations (the moves) and in the ultimate goal (checkmate); (2) it is neither so simple as to be trivial nor too difficult for satisfactory solution; (3) chess is generally considered to require "thinking" for skillful play; a solution of this problem will force us either to admit the possibility of a mechanized thinking or to further restrict our concept of "thinking"; (4) the discrete structure of chess fits well into the digital nature of modern computers.”
“Solved” Games

A game is solved if an optimal strategy is known.

Strong solved: *all positions.*
Weakly solved: *some (start) positions.*
Typical Game Setting

Games are usually:

- 2 player
- Alternating
- Zero-sum
  - Gain for one loss for another.
- Perfect information
Typical Game Setting

Very much like search:
- Set of possible states
- Start state
- Successor function
- Terminal states (many)
- Objective function

The key difference is alternating control.
Game Trees
Key Differences vs. Search

you select to max score

they select to min score

only get score here
Minimax

Propagate value backwards through tree.

\[
V(s0) = \max(V(s1), V(s2), V(s3))
\]

\[
V(s2) = \min(V(s4), V(s5), V(s6))
\]

\[
V(s5) = \max(V(g1), V(g2), V(g3))
\]
Minimax Algorithm

Compute value for each node, going backwards from the end-nodes.

Max (min) player: select action to maximize (minimize) return.

Optimal for both players (if zero sum).
Assumes perfect play, worst case.

Can run as depth first:
  • Time $O(b^d)$
  • Space $O(bd)$

Require the agent to evaluate the whole tree.
Minimax
Games of Chance

What if there is a chance element?
Stochasticity

An outcome is called *stochastic* when it is determined at random.
Stochasticity

How to factor in stochasticity?

Agent does not get to choose.
  • Selecting the max outcome is optimistic.
  • Selecting the min outcome is pessimistic.

Must be probability-aware.

Be aware of who is choosing at each level.
  • Sometimes it is you.
  • Sometimes it is an adversary.
  • Sometimes it is a random number generator.

insert randomization layer
ExpectiMax

You select (max) and they select to min score.
Expectation

**How to compute value of stochastic layer?**

What is the average die value?

\[
\frac{(1 + 2 + 3 + 4 + 5 + 6)}{6} = 3.5
\]

This factors in both probabilities and the value of event.

In general, given random event \( x \) and function \( f(x) \):

\[
E[f(x)] = \sum_x P(x) f(x)
\]
ExpectiMax

 stochastic (expectation)

 you select (max)

 stochastic (expectation)

 they select to min score

p1

dice

p1

...
Minimax

The diagram represents a minimax algorithm with the tree structure:

- **p1**
  - **max**
    - **p2**
      - **min**
        - 10
        - 5
      - **p2**
        - **min**
          - -3
          - 20
    - **p2**
      - **min**
        - -5
        - 2

The values at the leaves indicate the outcomes for each player. The algorithm aims to maximize the minimum possible outcome for the maximizing player and minimize the maximum possible outcome for the minimizing player.
In Practice

Can run as depth first:
  • Time $O(b^d)$
  • Space $O(bd)$

Depth is too deep.
  • 10s to 100s of moves.

Breadth is too broad.
  • Chess: 35, Go: 361.

Full search never terminates for non-trivial games.
What Is To Be Done?

Terminate early.
Branch less often.
Alpha-Beta

The diagram illustrates the Alpha-Beta pruning algorithm in a game tree. The tree is structured with nodes representing the game states, where each node has a value associated with it. The goal is to maximize the value for the maximizing player (max) and minimize the value for the minimizing player (min).

At each level, the algorithm prunes branches that cannot improve the best solution found so far, effectively reducing the search space and improving efficiency.

For example, in the subtree rooted at node p1, if the maximizing player (max) reaches a node with a value of 5, it knows that further exploration of this branch is unnecessary because a higher value is already guaranteed by the branch that leads to a value of 10, which was explored earlier and found to be the best choice.

This pruning process continues recursively, allowing the algorithm to focus on the most promising paths in the tree.
At a min layer:
If $V(B) \leq V(A)$ then prune B’s siblings.
At a max layer:
If $V(A) \geq V(B)$ then prune A’s siblings.
Alpha-Beta

More generally:
• $\alpha$ is highest max
• $\beta$ is lowest min

If max node:
• prune if $v \geq \beta$

If min node:
• prune if $v \leq \alpha$
function ALPHA-BETA-SEARCH(state) returns an action
    \( v \leftarrow \text{MAX-VALUE}(state, -\infty, +\infty) \)
    return the action in ACTIONS(state) with value \( v \)

function MAX-VALUE(state, \( \alpha, \beta \)) returns a utility value
    if TERMINAL-TEST(state) then return UTILITY(state)
    \( v \leftarrow -\infty \)
    for each \( a \) in ACTIONS(state) do
        \( v \leftarrow \text{MAX}(v, \text{MIN-VALUE}((\text{RESULT}(s, a), \alpha, \beta)) \)
        if \( v \geq \beta \) then return \( v \)
        \( \alpha \leftarrow \text{MAX}(\alpha, v) \)
    return \( v \)

function MIN-VALUE(state, \( \alpha, \beta \)) returns a utility value
    if TERMINAL-TEST(state) then return UTILITY(state)
    \( v \leftarrow +\infty \)
    for each \( a \) in ACTIONS(state) do
        \( v \leftarrow \text{MIN}(v, \text{MAX-VALUE}((\text{RESULT}(s, a), \alpha, \beta)) \)
        if \( v \leq \alpha \) then return \( v \)
        \( \beta \leftarrow \text{MIN}(\beta, v) \)
    return \( v \)

(from Russell and Norvig)
Alpha Beta Pruning

Single most useful search control method:
  • Throw away whole branches.
  • Use the min-max behavior.

Resulting algorithm: *alpha-beta pruning*.

Empirically: *square roots* branching factor.
  • Effectively doubles the search horizon.

Alpha-beta makes the difference between novice and expert computer game players. *Most successful players use alpha-beta.*
What Is To Be Done?

Terminate early.
Branch less often.
In Practice

Solution: substitute evaluation function.

- Like a heuristic - estimate value.
- In this case, probability of win or expected score.

- Common strategy:
  - Run to fixed depth then estimate.
  - Careful lookahead to depth $d$, then guess.
Evaluation Functions
Evaluation Functions
Deep Blue (1997)

480 Special Purpose Chips
200 million positions/sec
Search depth 6-8 moves (up to 20)
Evaluation Functions
Search Control

Horizon Effects

- What if something interesting at horizon + 1?
- How do you know?

More sophisticated strategies:

- When to generate more nodes?
- How to selectively expand the frontier?
- How to allocate fixed move time?
Monte Carlo Tree Search

Continually estimate value
Adaptively explore
Random rollouts to evaluate
Monte Carlo Tree Search

Step 1: path selection.
Monte Carlo Tree Search

Step 1: path selection.

\[ \frac{w_i}{n_i} + c \sqrt{\frac{\log n}{n_i}} \]  

UCT
Monte Carlo Tree Search

Step 2: expansion.
Monte Carlo Tree Search

Step 3: rollout.
Monte Carlo Tree Search

Step 4: update.
Games Today

World champion level:
• Backgammon
• Chess
• Checkers (solved)
• Othello
• Some poker types:

Perform well:
• Bridge
• Other poker types

Far off: Go
Very Recently

Lee Sedol

1 - 4

AlphaGo (Google Deepmind)
Board Games

“... board games are more or less done and it's time to move on.”