Game Playing

In game playing, the choice of action must take the opponent into account.

A search problem for a game is defined by:

Initial state. Current position and whose turn.

Operators. The legal moves.

Terminal test. Is the game over in a given state?

Utility function. Is a terminal state win, lose, or draw?

Notation

MAX. The player whose turn it is to move.

MIN. The other player.

Ply. A synonym for "depth."

Evaluation function. Estimates MAX's utility.

Minimax Search

MAX should maximize utility/evaluation assuming that MIN minimizes utility/evaluation.

function MAX-VALUE(*state*, *bound*)
  if TERMINAL(*state*) or *bound* $= 0$
    then return EVALUATION(*state*)
  $eval \leftarrow -\infty$
  for each *next* in EXPAND(*state*)
    do $eval \leftarrow \max(eval,$ MIN-VALUE($next, bound - 1$))
  return *eval*

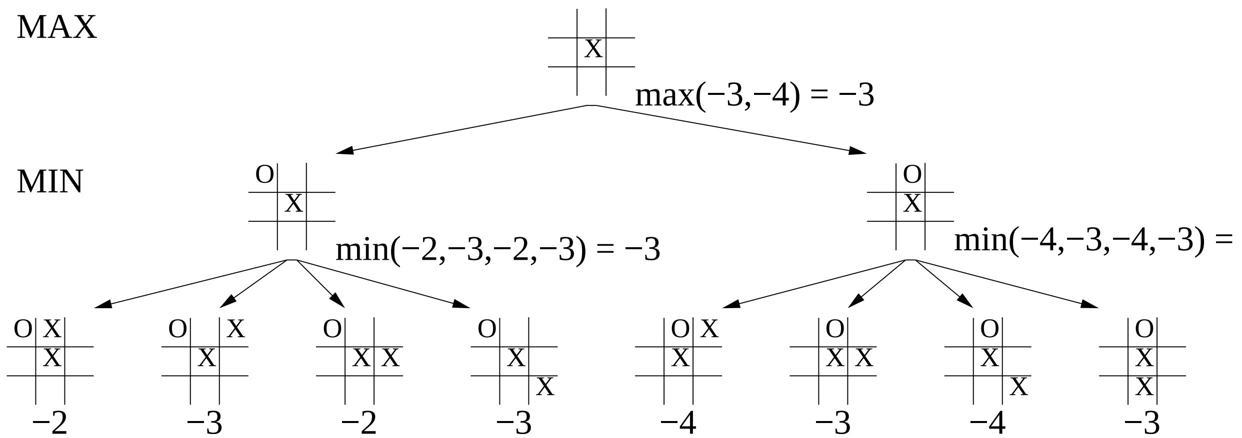
---

function MIN-VALUE(*state*, *bound*)
  if TERMINAL(*state*) or *bound* $= 0$
    then return EVALUATION(*state*)
  $eval \leftarrow \infty$
  for each *next* in EXPAND(*state*)
    do $eval \leftarrow \min(eval,$ MAX-VALUE($next, bound - 1$))
  return *eval*

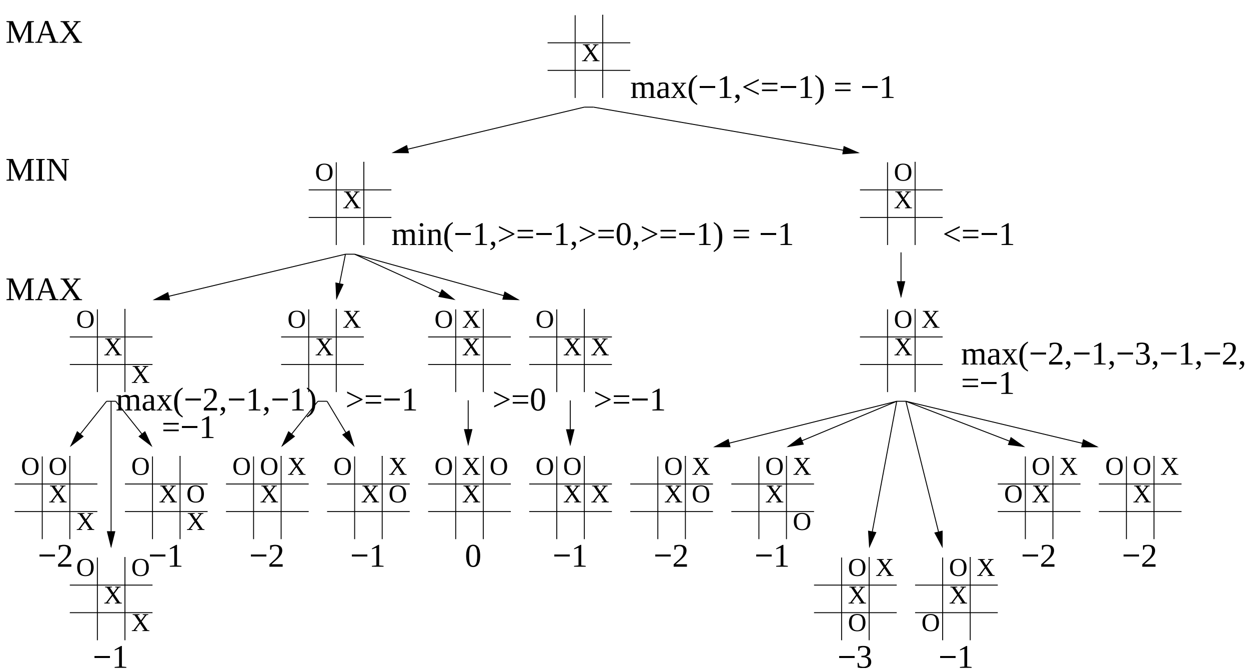
Minimax

evaluation = +10 if I have 3 in a row
−10 if my opponent has 3 in a row
+1 for each potential 3 in a row for me
−1 for each potential 3 in a row for my opponent

MAX is O; MIN is X



Minimax with Alpha−Beta Pruning



Alpha-Beta Pruning

Stop search when no effect is detected.
α is a known maximum value.
β is a known minimum value.

function MAX-VALUE(*state*, *bound*, α, β)
  if TERMINAL(*state*) or *bound* $= 0$
    then return EVALUATION(*state*)
  for each *next* in EXPAND(*state*)
    do $\alpha \leftarrow \max(\alpha,$ MIN-VALUE(*next*,*bound*-1,α,β))
      if $\alpha \geq \beta$ **then return** α
  return α

---

function MIN-VALUE(*state*, *bound*, α, β)
  if TERMINAL(*state*) or *bound* $= 0$
    then return EVALUATION(*state*)
  for each *next* in EXPAND(*state*)
    do $\beta \leftarrow \min(\beta,$ MAX-VALUE(*next*,*bound*-1,α,β))
      if $\beta \leq \alpha$ **then return** β
  return β

Performance of Minimax and Alpha-Beta

b = branching factor
d = depth of search

Minimax visits every state from level 0 to d.

$$\sum_{i=0}^{d} b^i = \frac{b^{d+1} - 1}{b - 1} \in O(b^d)$$

Alpha-Beta visits as few as $\Omega(b^{d/2})$ states.
Depends on a good ordering from EXPAND.
Actual programs approach the minimum bound.

Allows programs to look ahead twice as many moves.

Other Issues:
Horizon problem
Quiescence
Data bases of openings and end games
Games of chance