NIM - a two person game
• n objects are in one pile
• each player divides a pile into two unequal piles; this continues until no more divisions are possible
• The loser is the person who cannot make any more legal moves

Let’s Play NIM
• Start with the 7 objects shown on the previous slide. Devise a winning strategy so that you will win no matter what choices are made by your opponent.
• Please note that you have the right to select NOT going first
A Minimax approach to NIM

- We alternate moves between players labeled Min and Max; Max is the player trying to win and Min is the opponent attempting to minimize Max’s score.
- Since we can search clear to leaves, we label wins for Min with 0 and wins for Max with 1; we propagate moves up the tree.
  - Max always gets the max of the children.
  - Min always gets the min of the children.
- Who can win this game?

Another game of NIM

- Start with 8 initial objects.
- Draw the complete search space.
- Label the initial node as Max and calculate the values for each state.
- Determine if there is a winning strategy for either player.
Using a fixed ply depth

- Most games are so complex you cannot analyze the entire search space, rather you look a fixed number of plys ahead.
- You start at the leaves and work backwards maximizing or minimizing as appropriate.
- The value at the start node will tell the player which move to take.

Minimax for Tic-tac-toe

- The heuristic attempts to measure the conflict between the two players, namely, the number of winning lines open to max minus the number of lines open to min.
- A forced win for max evaluates to $+\infty$ and a forced win for min to $-\infty$.

Heuristic $E(n) = M(n) - O(n)$

where $M(n)$ is the total of Max's possible winning lines
$O(n)$ is total of Opponent's possible winning line
$E(n)$ is the total Evaluation for state $n$.
The start game

- The first two plys are shown
- The choices for max are -1 to the left, -2 in the center, or +1 on the right
- clearly the right path (mark in the center) is superior

After the first two moves

- Again we go two plys deep
- Each leaf state has been evaluated and the values propagated up the tree
- The best move for max is the lower left corner
Continuing the game

- O blocks X by marking the upper right corner; every alternative for X loses except to block O by marking the upper left corner.
- Clearly this will be a winning move for X since two squares can produce 3 in a row and O can only block one of them.

Another Minimax Search

- Perform a minimax search on this tree.
Alpha-beta pruning

- As the minimax search proceeds, it is possible to discontinue to search at certain points when no better results are possible
- We will cover alpha-beta pruning

A pruned Minimax tree

A has $\beta = 3$ (A will be no larger than 3)
B is $\beta$ pruned, since $5 > 3$
C has $\alpha = 3$ (C will be no smaller than 3)
D is $\alpha$ pruned, since $0 < 3$
E is $\alpha$ pruned, since $2 < 3$
C is 3
To MINIMAX with ALPHA-BETA:
1. Determine if the level is the top level, or if the limit of search has been reached, or if the level is a minimizing level, or if the level is a maximizing level:
   1a. If the level is the top level, let alpha be $-\infty$ and let beta be $+\infty$.
   1b. If the limit of search has been reached, compute the static value of the current position relative to the appropriate player. Report the result.
1c. If the level is a minimizing level:
   1c1. Until all children are examined with MINIMAX or alpha is bigger than beta:
      1c1.1. Set beta to the smaller of the given beta values and the smallest value so far reported by MINIMAX working on the children.
      1c1.2. Use MINIMAX on the next child of the current position, handing this new application of MINIMAX the current alpha and beta.
1c2. Report beta.
1d. If the level is a maximizing level:
1d1. Until all children are examined with MINIMAX or alpha is bigger than beta:
   1d1.1. Set alpha to the larger of the given alpha values and the biggest value so far reported by MINIMAX working on the children.
   1d1.2. Use MINIMAX on the next child of the current position, handing this new application of MINIMAX the current alpha and beta.
1d2. Report alpha.

Practicing alpha beta pruning

- Perform a minimax search with alpha beta pruning on this tree

[Diagram of a minimax tree with nodes labeled A to M and numbers 3 to 8 at the leaves.]

Samuel’s Checkers Program

• The first (1959) significant program involving a two opponent game
• There was a complex heuristic $\sum a_i x_i$ that summed up the weighted value of various individual values based on piece advantage, piece location, control of center, etc.
• If the selected move resulted in a loss, the highest weight values were “blamed” and their weights reduced; this, in effect, implemented a simple learning algorithm
• Since the approach was basically hill climbing, the program was susceptible to being lead into traps

Complexity issues

• Let $B$ be the average branching factor and $L$ the path length, then the total number of states is $T = B + B^2 + B^3 + \ldots + B^L = B (B^{L-1} - 1) (B - 1)$
• As seen in the graph below, the value of $B$ has a dramatic effect on the overall complexity
Variation in informedness

- In general, if a heuristic is more informed
  - it will be more complex to compute
  - it will reduce the rule application cost by cutting down the search space
- Summing up these two conflicting values produces a graph with a minimum that represents the problem solving cost

Think of a Game

- Think of a game where it is possible to have simple strategies or very complex strategies
- Assume you have a fixed amount of time before being forced to make a decision
- Would the most complex strategy be the best strategy? If not, why not?