Part 2: Implement heuristic
  - Most constrained variable
  - Least constraining value
  - Forward checking
  - Combination
    - Your own

I write in customSolver()
Adversarial Environments / Game Playing

So far all environments have been benign (no opponents)

Tic Tac Toe as Search:

Suppose playing against an opponent.
We go first.

\[
\begin{array}{ccc}
\times & \times & \times \\
\times & 0 & 0 \\
0 & 1 & 1 \\
\end{array}
\]

utility: +1 0 -1

Don't just want any leaf, but best one!
Assign each value an utility.

utility - measures the value of a state

Goal - node with highest utility
Back to T^3:

Suppose

\[
\begin{array}{ccc}
X & X & 0 \\
0 & X & 0 \\
X & & \\
\end{array}
\]

0's turn

\[
\begin{array}{ccc}
X & X & 0 \\
0 & X & 0 \\
0 & & 10 \\
\end{array}
\]

Which will O do? Don't know but we should play to minimize risk - assume opponent will always be optimal for him/her.

Optimal strategy - yields optimal solution regardless of opponent's moves.

Consider designing such a tree for another game.

Nodes ~ states

Edges ~ players' moves

X's turn \{ 2 moves left \}

0's turn

Suppose we are X.

How can X play optimally? Simplest?

Greedy.
Greedy strategy - choose the action/move that yields
state with highest utility.

ex: X chooses B.

Problem: O chooses C.

minimize risk - assume opponent's move will always be
optimal for him/her.

Best choice for X? Choose C.

Minimax Principle
Assumption: both players play optimally
MAX (x): tries to maximize utility
MIN (o): " minimize MAX's utility

MIN - adversary.

How did we know C was best choice?
" start at leaves
" propagate utility upwards:
- if MIN's turn: choose action that minimizes MAX's utility
  " MAX's ": " " maximizes " " "

ex: O's (MIN) turn: -7, -6, 0

X's (MAX) turn: 0
ex 2

What kind of search to find MAX's utility value?

DEF:

\[
\text{minimax (State)}
\]

if state is leaf:

return utility(state)

if MAX's turn:

Let \( c_1, c_2, \ldots, c_k \) denote children of current state

return \( \max_i(\text{minimax}(c_i)) \)

if MIN's turn:

let \( c_1, \ldots, c_k \)

return \( \min(\text{minimax}(c_1), \ldots) \)

Analyze:

Assume finite state space

\( O(b^d) \):

1. Space: \( O(b^d) \).
2. Time: \( b = \max \# \text{legal moves from any state} \)
   \[ d = \max \text{depth} / \# \text{moves} \]
Complete & Optimal

Optimal: Always maximize utility against any opponent?

Question: What if opponent does not play optimally?
MAX will get even higher utility.

Baby Nim example.
- Is there a winning strategy for MAX?
- If so, what is it (which moves?)

Solution: Prune the tree.

Another example:

```
 +---+
 | A |        MAX
 |   |        |
 |   |        |   
 +---+        +---+
     D        +---+
     4        | A |
     12        |   |
     7        +---+        MIN
     10        | B |
     3        +---+        |
     16        |   |
     2        +---+        MIN
     4        | C |
     1        +---+        |
```

Branch A

MIN: 4 first, then 12, then 7. ⇒ min = 4

What does this imply about how much MAX can get?
MAX will get ≥ 4.
Baby Nim

MAX

1

MIN

2

MAX

(3)

MIN

(2)

MAX

(1)

+1