Adversarial Search

CS51A
David Kauchak
Spring 2022

Some material borrowed from :
Sara Owsley Sood and others

1

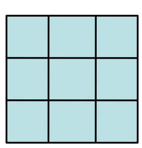
Admin

Assignment 10

Midterm 2 on Monday
- From dictionaries (2/21) through informed search (4/11)
- Practice problems available
- 2 page “cheat” sheet
- Will try and have an additional mentor session for midterm questions over the weekend

2

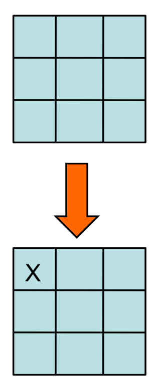
Tic Tac Toe as search



Given a state (i.e. board configuration), what move should we make!

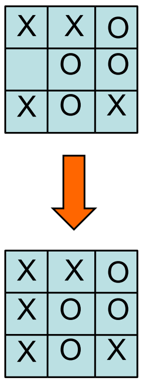
3

Tic Tac Toe as search



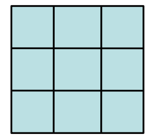
4

Tic Tac Toe as search



5

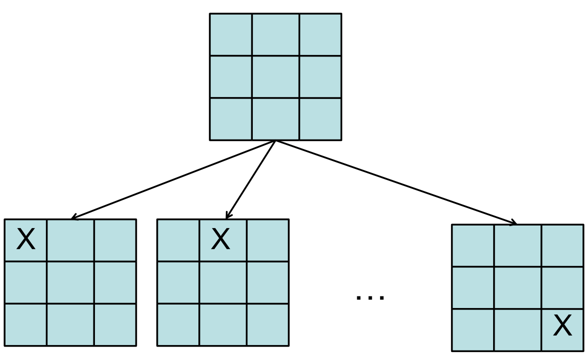
Tic Tac Toe as search



How can we pose this as a search problem?

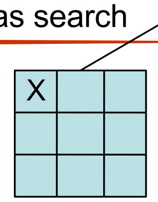
6

Tic Tac Toe as search



7

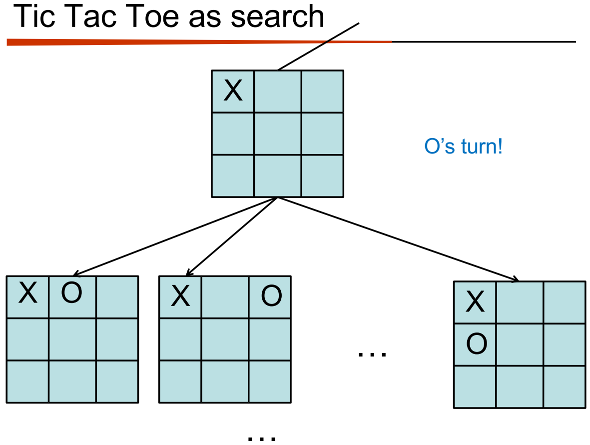
Tic Tac Toe as search



Now what?

8

Tic Tac Toe as search

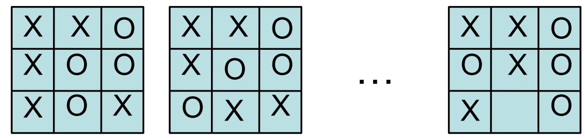


O's turn!

9

Tic Tac Toe as search

Eventually, we'll get to a state without any options



WIN TIE LOSE

Try and make moves that move us towards a win, i.e. where there are leaves with a WIN.

10

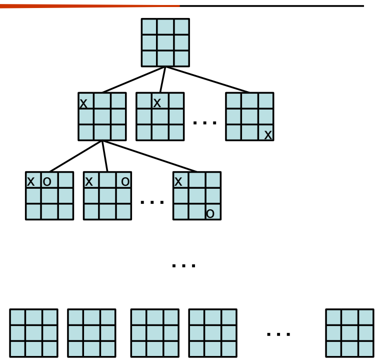
Tic Tac Toe

X's turn

O's turn

X's turn

...

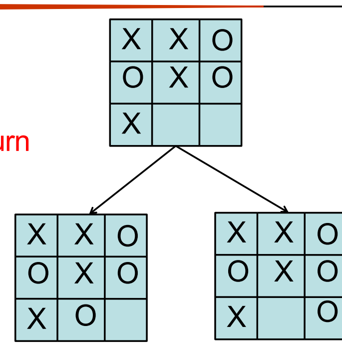


Problem: we don't know what O will do

11

I'm X, what will 'O' do?

O's turn



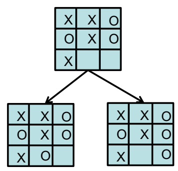
12

Minimizing risk

The computer doesn't know what move O (the opponent) will make

It can *assume* that it will try and make the best move possible

Even if O actually makes a different move, we're no worse off. Why?



13

Optimal Strategy

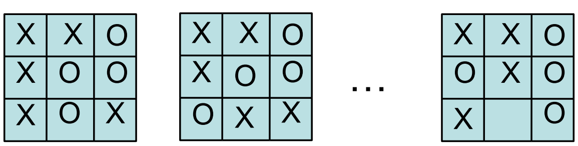
An Optimal Strategy is one that is at least as good as any other, no matter what the opponent does

– If there's a way to force the win, it will

– Will only lose if there's no other option

14

Defining a scoring function



WIN +1

TIE 0

LOSE -1

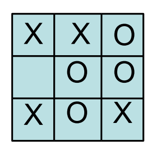
Idea:

- define a function that gives us a "score" for how good each state is
- higher scores mean better

15

Defining a scoring function

Our (X) turn



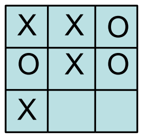
What should be the score of this state?

+1: we can get to a win

16

Defining a scoring function

Opponent's (O) turn



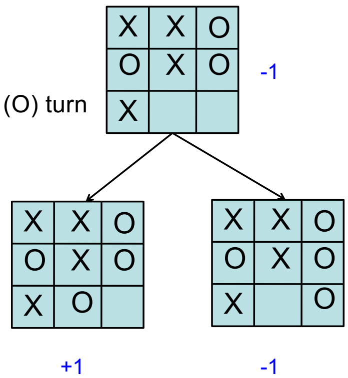
What should be the score of this state?

-1: opponent can get to a win

17

Defining a scoring function

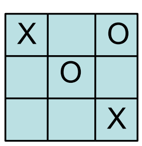
Opponent's (O) turn



18

Defining a scoring function

Our (X) turn



What should be the score of this state?

19

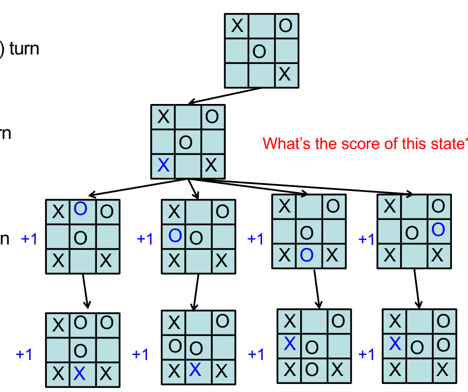
Defining a scoring function

Our (X) turn

O turn

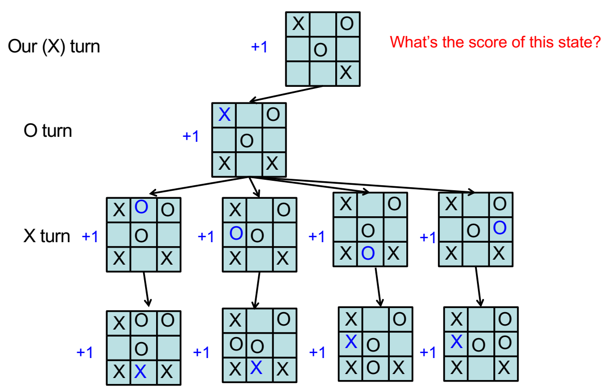
What's the score of this state?

X turn



20

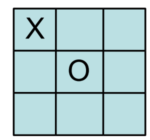
Defining a scoring function



21

Defining a scoring function

Our (X) turn

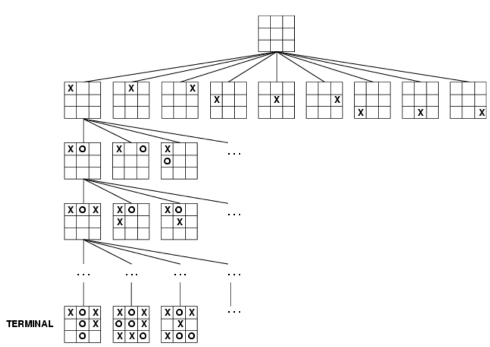


What should be the score of this state?

0: If we play perfectly and so does O, the best we can do is a tie (could do better if O makes a mistake)

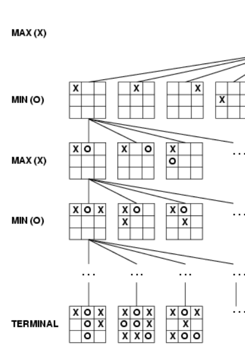
22

How can X play optimally?



23

How can X play optimally?



When it's my turn, pick the highest scoring state

When it's the opponent's turn, assume the lowest scoring state (from my perspective)

If we can reach the end games, we can percolate these answers all the way back up

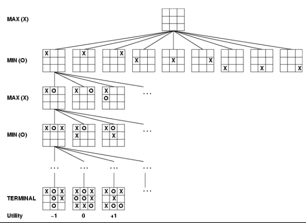
24

How can X play optimally?

Start from the bottom and propagate the score up:

- if X's turn, pick the move that maximizes the utility
- if O's turn, pick the move that minimizes the utility

Is this optimal?



25

Minimax Algorithm: An Optimal Strategy

minimax(state) =
    if state is a terminal state
        score(state)
    else if MY turn
        over all next states, s: return the *maximum* of minimax(s)
    else if OPPONENTS turn
        over all next states, s: return the *minimum* of minimax(s)

Uses recursion to compute the "value" of each state

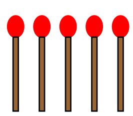
Searches down to the leaves, then the values are "backed up" through the tree as the recursion finishes

What type of search is this?

What does this assume about how MIN will play? What if this isn't true?

26

Baby Nim

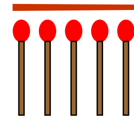


Take 1 or 2 at each turn
Goal: take the last match

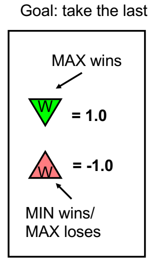
What move should I take?

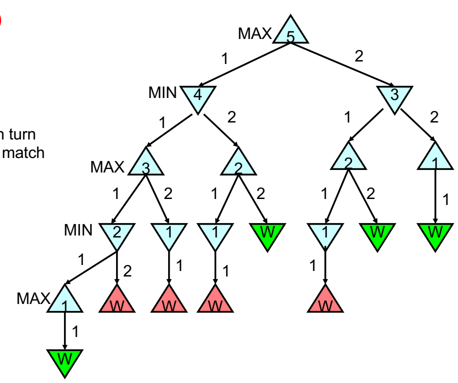
27

Baby Nim

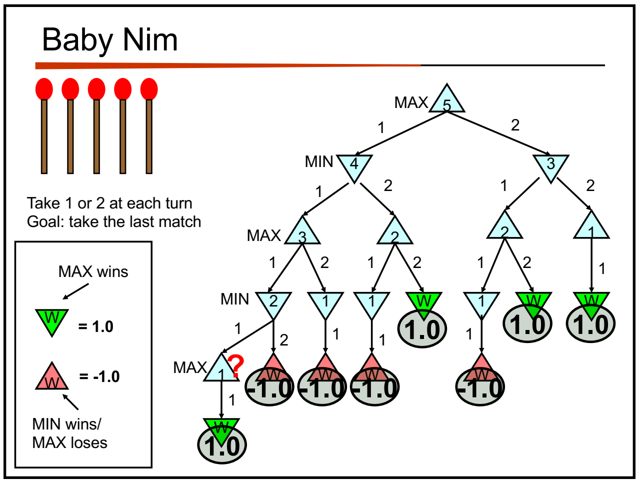


Take 1 or 2 at each turn
Goal: take the last match

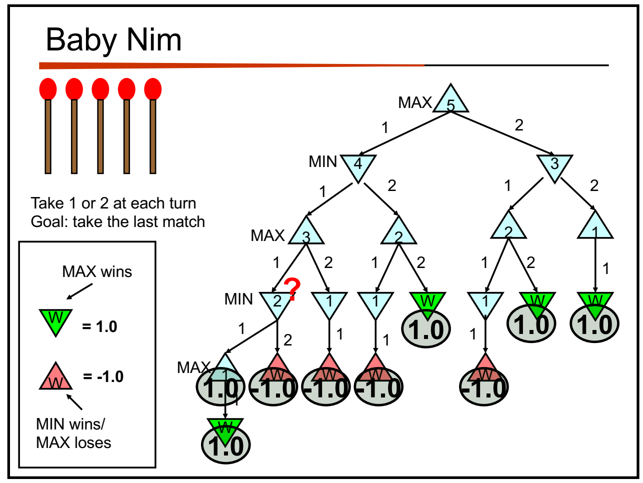




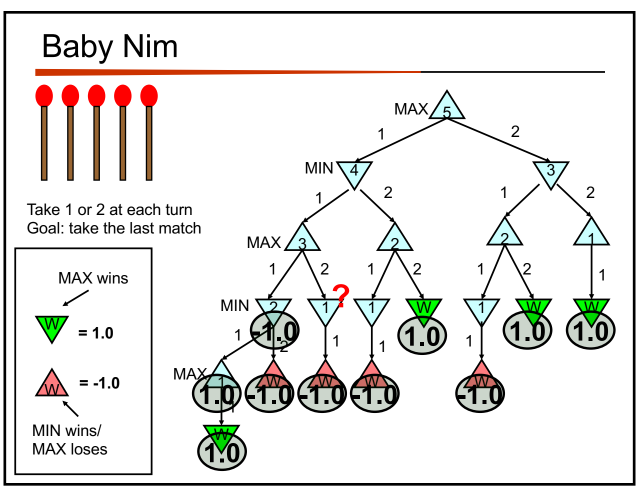
28


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses
MAX
MIN

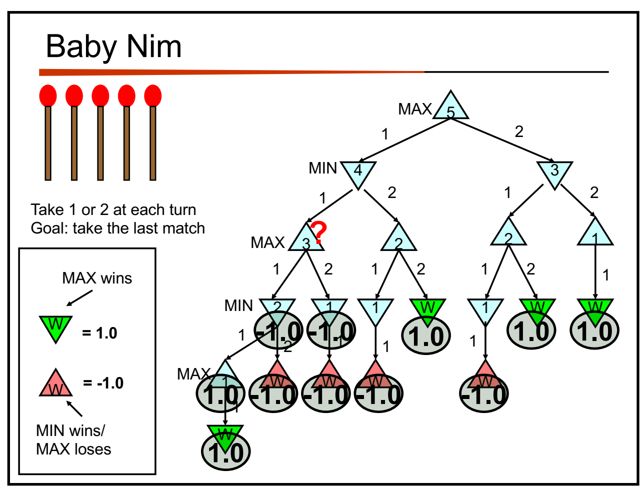
29


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses
MAX
MIN

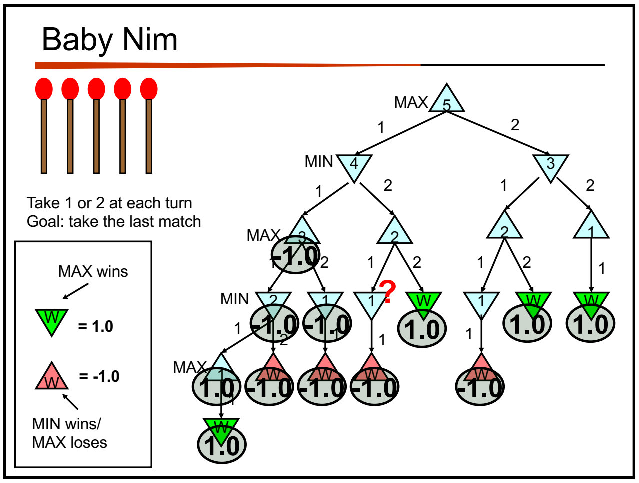
30


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses
MAX
MIN

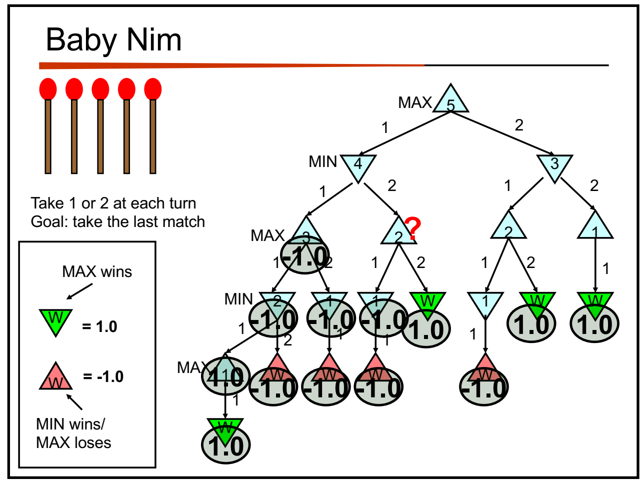
31


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses
MAX
MIN

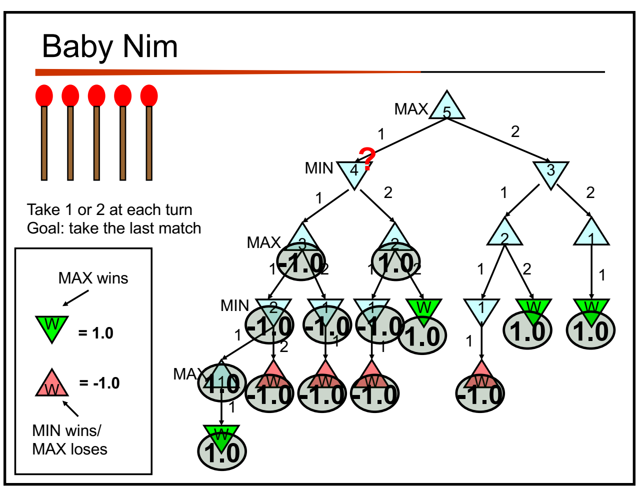
32


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

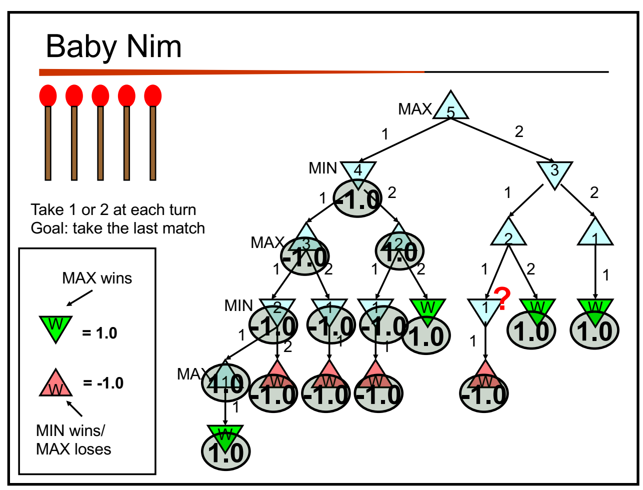
33


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

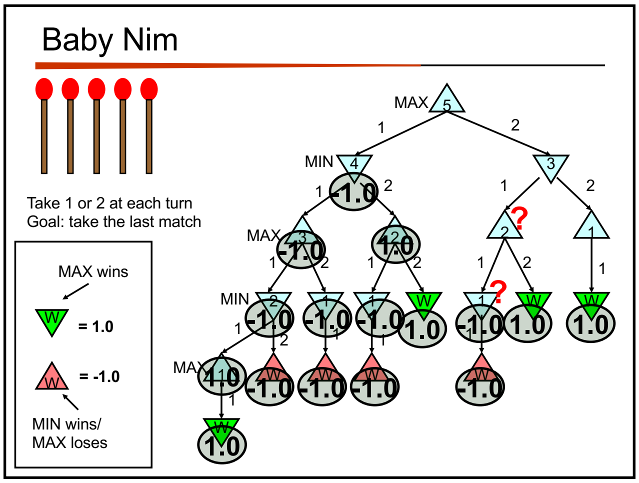
34


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

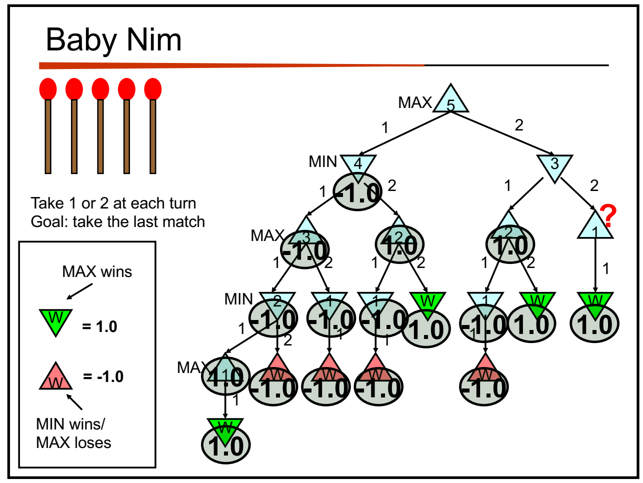
35


Baby Nim
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

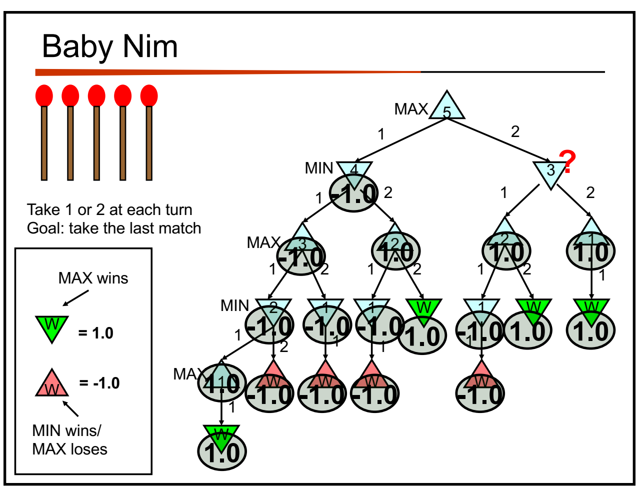
36


Baby Nim
MAX
MIN
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

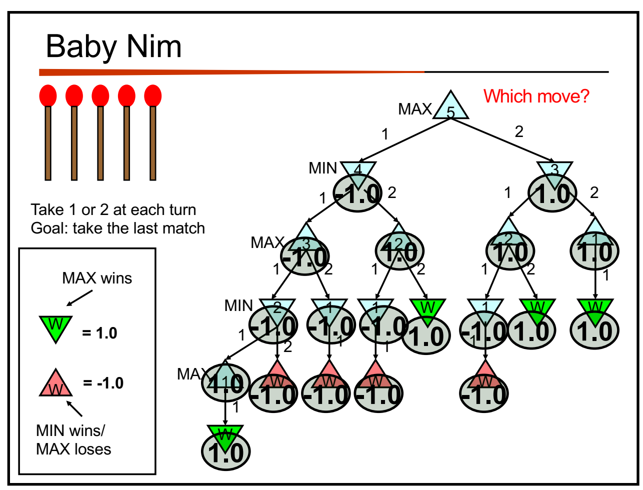
37


Baby Nim
MAX
MIN
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

38

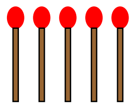

Baby Nim
MAX
MIN
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

39


Baby Nim
Which move?
MAX
MIN
Take 1 or 2 at each turn
Goal: take the last match
MAX wins
= 1.0
= -1.0
MIN wins/
MAX loses

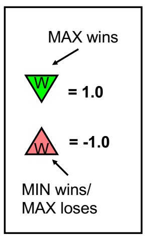
40

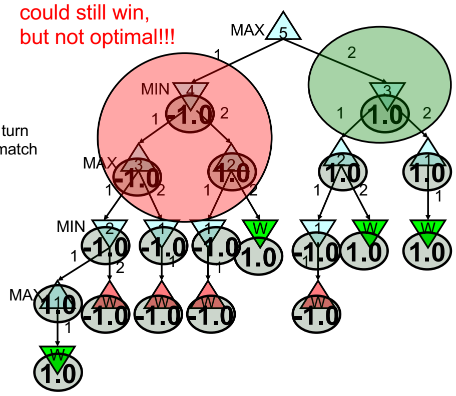
Baby Nim



could still win,
but not optimal!!!

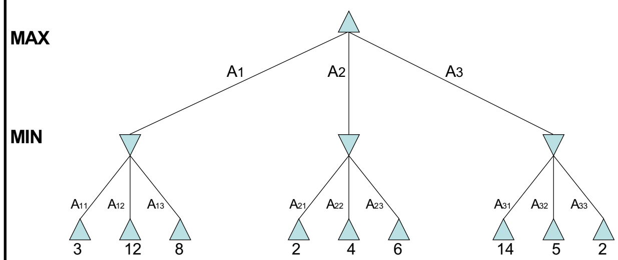
Take 1 or 2 at each turn
Goal: take the last match





41

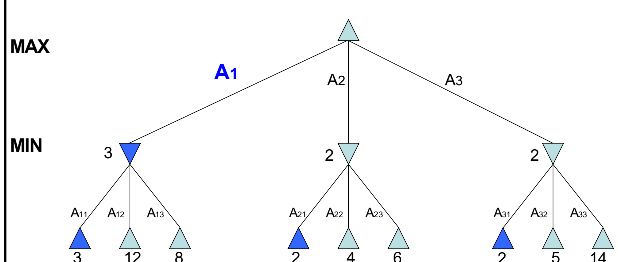
Minimax example 2



Which move should be made: A_1, A_2 or A_3?

42

Minimax example 2



43

Properties of minimax

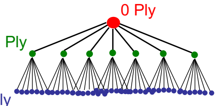
Minimax is optimal!

Are we done?

44

Games State Space Sizes

On average, there are ~35 possible moves that a chess player can make from any board configuration…



18 Ply!!

17005

Hydra at home in the United Arab Emirates…

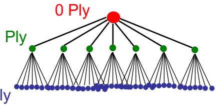
Branching Factor Estimates for different two-player games

Tic-tac-toe	4
Connect Four	7
Checkers	10
Othello	30
Chess	35
Go	300

45

Games State Space Sizes

On average, there are ~35 possible moves that a chess player can make from any board configuration…



Boundaries for qualitatively different games…

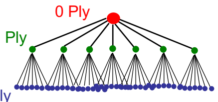
Branching Factor Estimates for different two-player games

Tic-tac-toe	4
Connect Four	7
Checkers	10
Othello	30
Chess	35
Go	300

46

Games State Space Sizes

On average, there are ~35 possible moves that a chess player can make from any board configuration…



Can search entire space

"solved" games

CHINOOK (2007) →

computer-dominated

Can't ☹

Is this true? human-dominated

Branching Factor Estimates for different two-player games

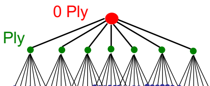
Tic-tac-toe	4
Connect Four	7
Checkers	10
Othello	30
Chess	35
Go	300

47

Games State Space Sizes

AlphaGo (created by Google), in April 2016 beat one of the best Go players:

http://www.nytimes.com/2016/04/05/science/google-alphago-artificial-intelligence.html



Can search entire space

"solved" games

CHINOOK (2007) →

computer-dominated

Can't ☹

What do we do?

Branching Factor Estimates for different two-player games

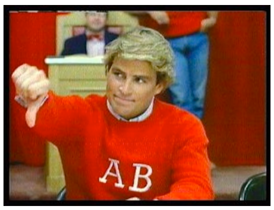
Tic-tac-toe	4
Connect Four	7
Checkers	10
Othello	30
Chess	35
Go	300

48

Alpha-Beta pruning

An optimal pruning strategy

- only prunes paths that are suboptimal (i.e. wouldn't be chosen by an optimal playing player)
- returns the *same* result as minimax, but faster



Name the movie ☺

49

Alpha-Beta pruning

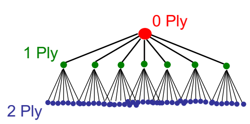
50

Games State Space Sizes

Pruning helps get a bit deeper

For many games, still can't search the entire tree

Now what?



Branching Factor Estimates
for different two-player games

Tic-tac-toe	4
Connect Four	7
Checkers	10
Othello	30
Chess	35
Go	300

computer-dominated

51

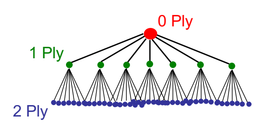
Games State Space Sizes

Pruning helps get a bit deeper

For many games, still can't search the entire tree

Go as deep as you can:

- *estimate* the score/quality of the state (called an evaluation function)
- use that instead of the real score



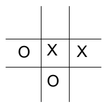
Branching Factor Estimates
for different two-player games

Tic-tac-toe	4
Connect Four	7
Checkers	10
Othello	30
Chess	35
Go	300

computer-dominated

52

Tic Tac Toe evaluation functions



Ideas?

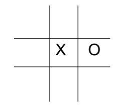
53

Example Tic Tac Toe EVAL

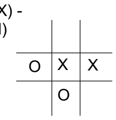
Tic Tac Toe
Assume MAX is using "X"

EVAL(*state*) =

if state is win for MAX:
+ ∞
if state is win for MIN:
- ∞
else:
(number of rows, columns and diagonals available to MAX) -
(number of rows, columns and diagonals available to MIN)



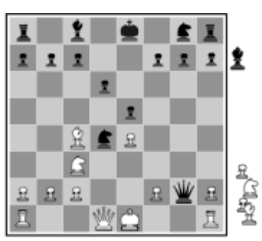
= 6 - 4 = 2



= 4 - 3 = 1

54

Chess evaluation functions



Ideas?

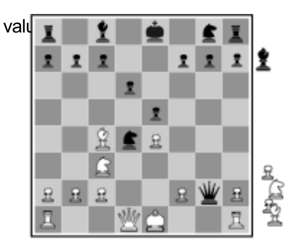
55

Chess EVAL

Assume each piece has the following valu

pawn	= 1;
knight	= 3;
bishop	= 3;
rook	= 5;
queen	= 9;

EVAL(*state*) =
sum of the value of white pieces –
sum of the value of black pieces



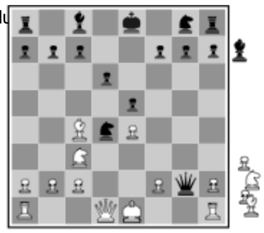
= 31 - 36 = -5

56

Chess EVAL

Assume each piece has the following valu
- pawn = 1;
- knight = 3;
- bishop = 3;
- rook = 5;
- queen = 9;

EVAL(*state*) =
sum of the value of white pieces –
sum of the value of black pieces



Any problems with this?

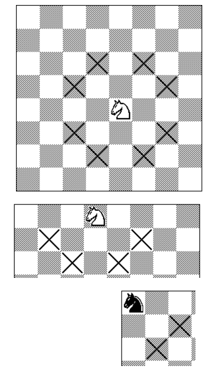
57

Chess EVAL

Ignores actual positions!

Actual heuristic functions are often a weighted combination of features

$$EVAL(s) = w_1 f_1(s) + w_2 f_2(s) + w_3 f_3(s) + \ldots$$



58

Chess EVAL

$$EVAL(s) = w_1 f_1(s) + w_2 f_2(s) + w_3 f_3(s) + \ldots$$

- f_1: number of pawns
- f_2: number of attacked knights
- f_3: 1 if king has knighted, 0 otherwise

A feature can be any numerical information about the board
- as general as the number of pawns
- to specific board configurations

Deep Blue: 8000 features!

59

history/end-game tables

History
- keep track of the quality of moves from previous games
- use these instead of search

end-game tables
- do a reverse search of certain game configurations, for example all board configurations with king, rook and king
- tells you what to do in ***any*** configuration meeting this criterion
- if you ever see one of these during search, you lookup exactly what to do

60

end-game tables

Devastatingly good

Allows much deeper branching
- for example, if the end-game table encodes a 20-move finish and we can search up to 14
- can search up to depth 34

Stiller (1996) explored all end-games with 5 pieces
- one case check-mate required 262 moves!

Knoval (2006) explored all end-games with 6 pieces
- one case check-mate required 517 moves!

Traditional rules of chess require a capture or pawn move within 50 or it's a stalemate

61

Opening moves

At the very beginning, we're the farthest possible from any goal state

People are good with opening moves

Tons of books, etc. on opening moves

Most chess programs use a database of opening moves rather than search

62

Nim

K piles of coins

On your turn you must take one or more coins from one pile

Player that takes the last coin wins

Example:
https://www.goobix.com/games/nim/

63