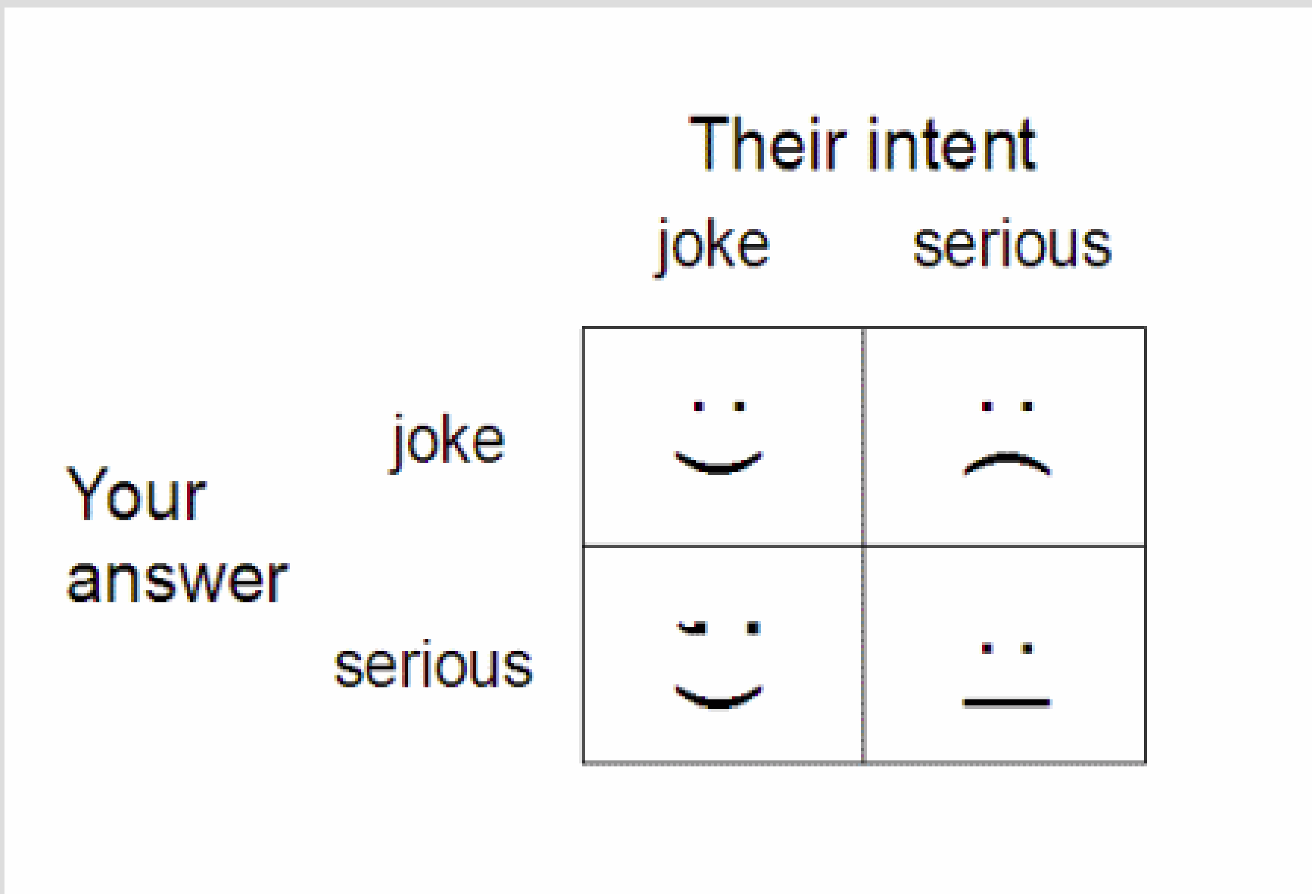
Game theory (Ch. 17.5)



MCTS

How to find which actions are “good”?

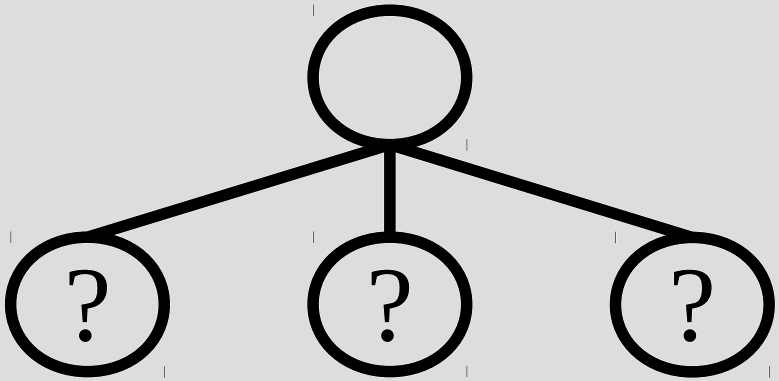
The “Upper Confidence Bound applied to Trees” UCT is commonly used:

$$\max\left(\frac{win(n)}{times(n)} + \sqrt{\frac{2\ln TotalTimes}{times(n)}}\right)$$

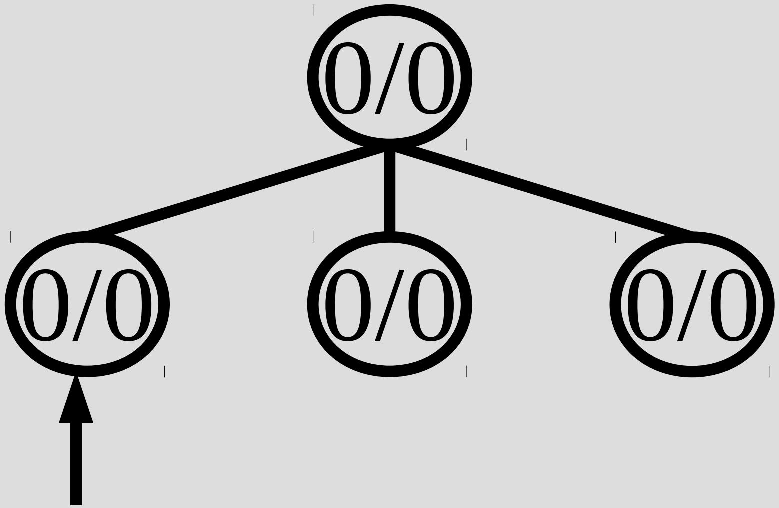
This ensures a trade off between checking branches you haven't explored much and exploring hopeful branches

(https://www.youtube.com/watch?v=Fbs4lnGLS8M)

MCTS



MCTS

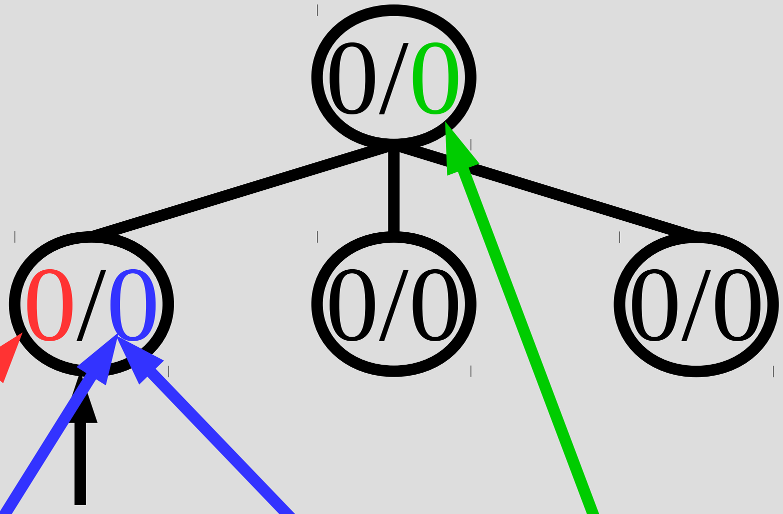


$$\frac{win(n)}{times(n)} + \sqrt{\frac{2\ln TotalTimes}{times(n)}}$$

$$= \frac{0}{0} + \sqrt{\frac{2ln0}{0}}$$

$$= \infty$$

MCTS

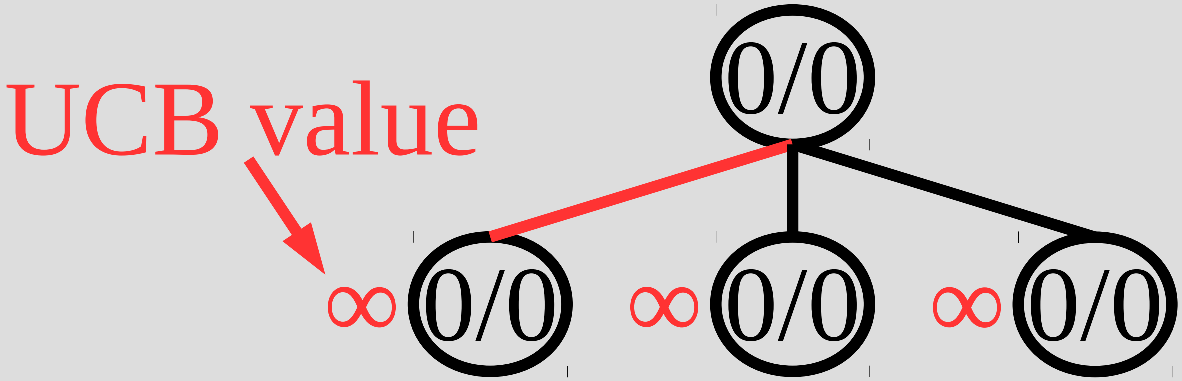


$$\frac{win(n)}{times(n)} + \sqrt{\frac{2\ln TotalTimes}{times(n)}}$$

$$= \frac{0}{0} + \sqrt{\frac{2ln0}{0}}$$

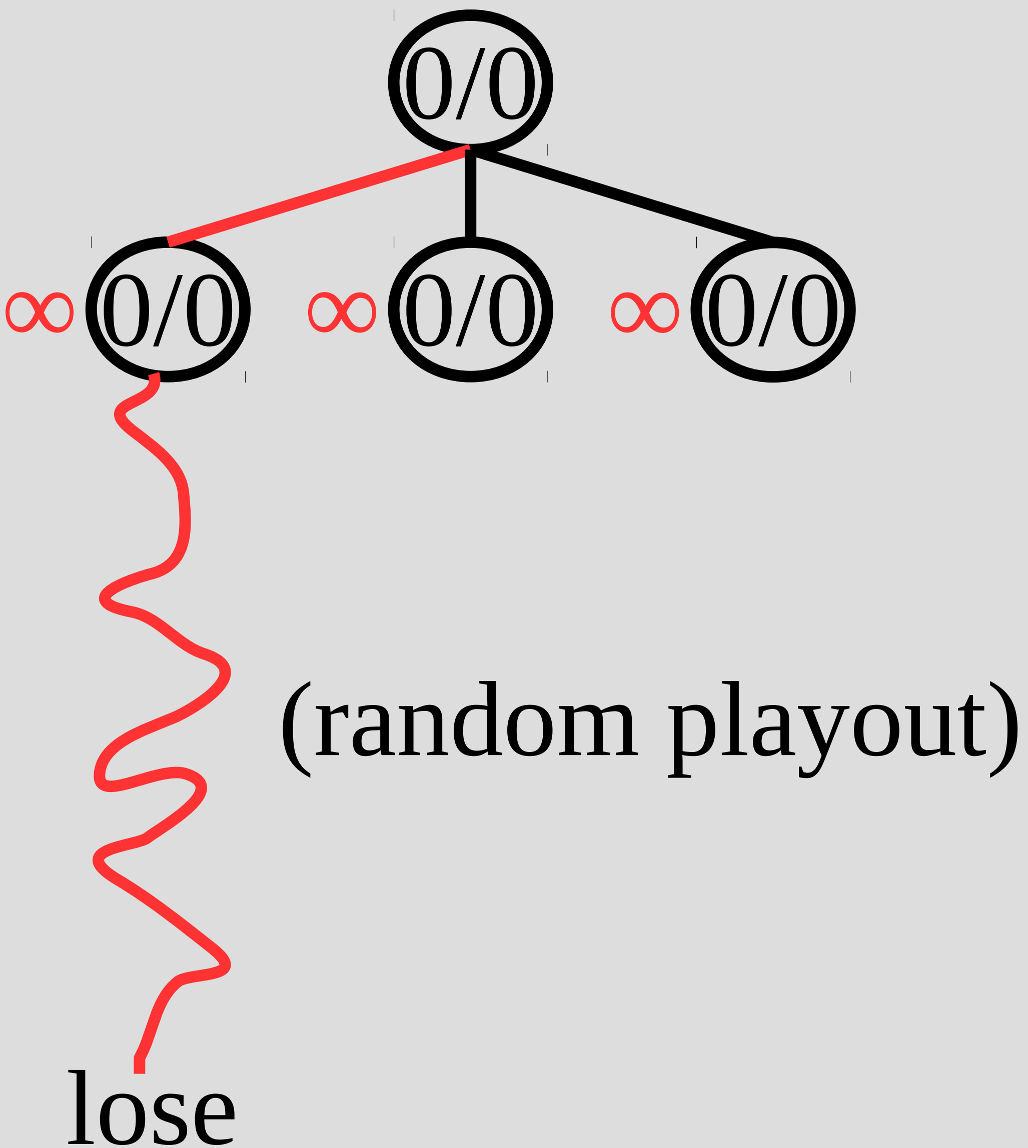
$$= \infty$$

MCTS

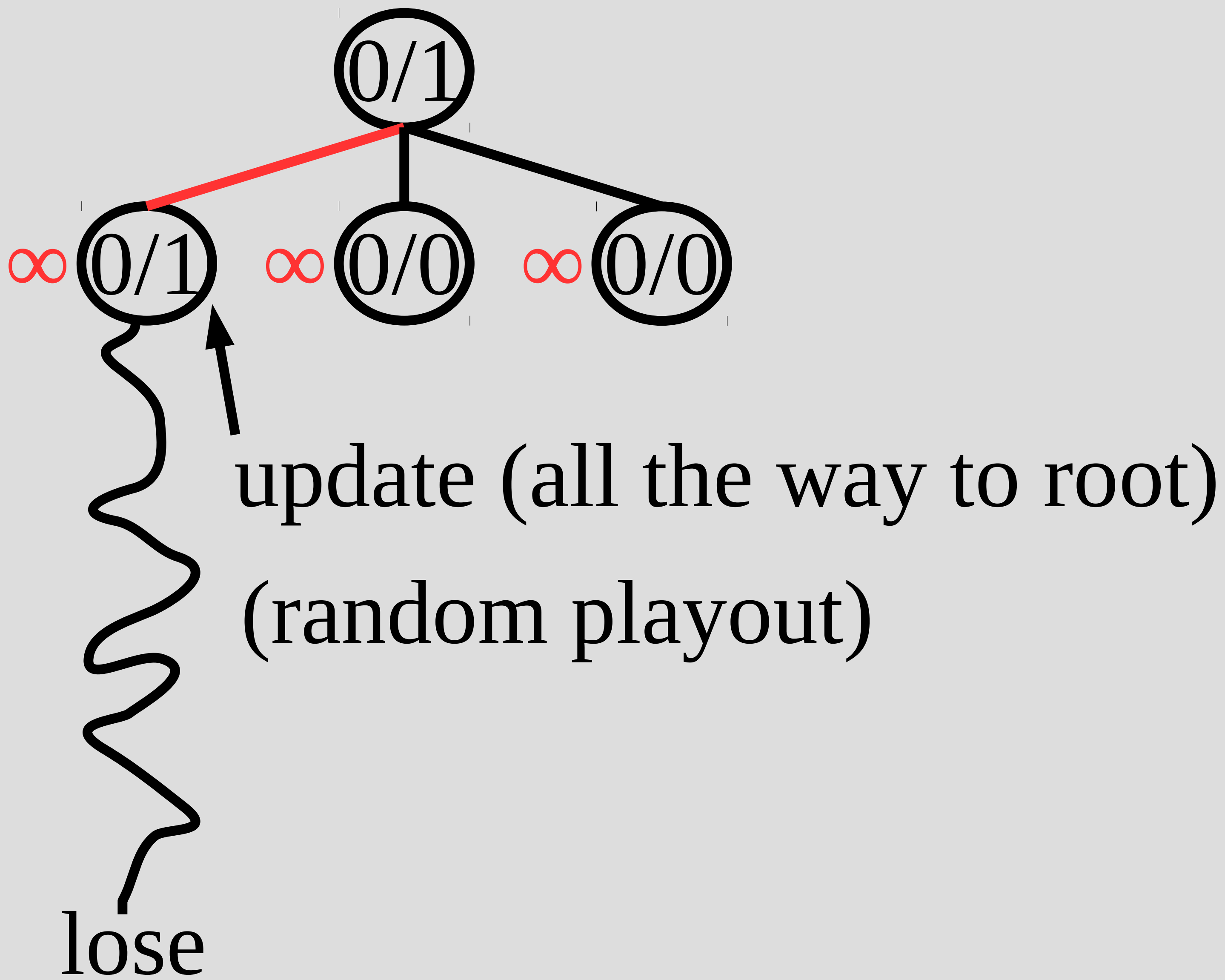


Pick max (I'll pick left-most)

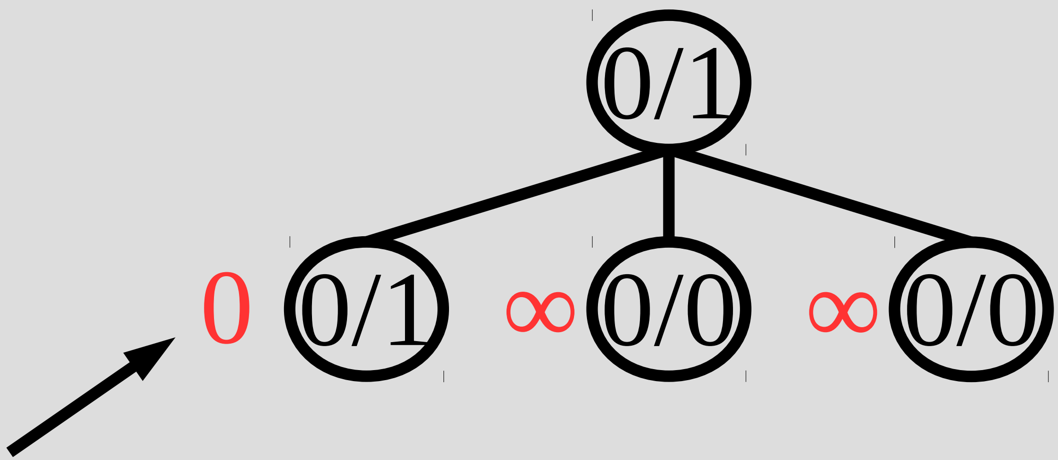
MCTS



MCTS



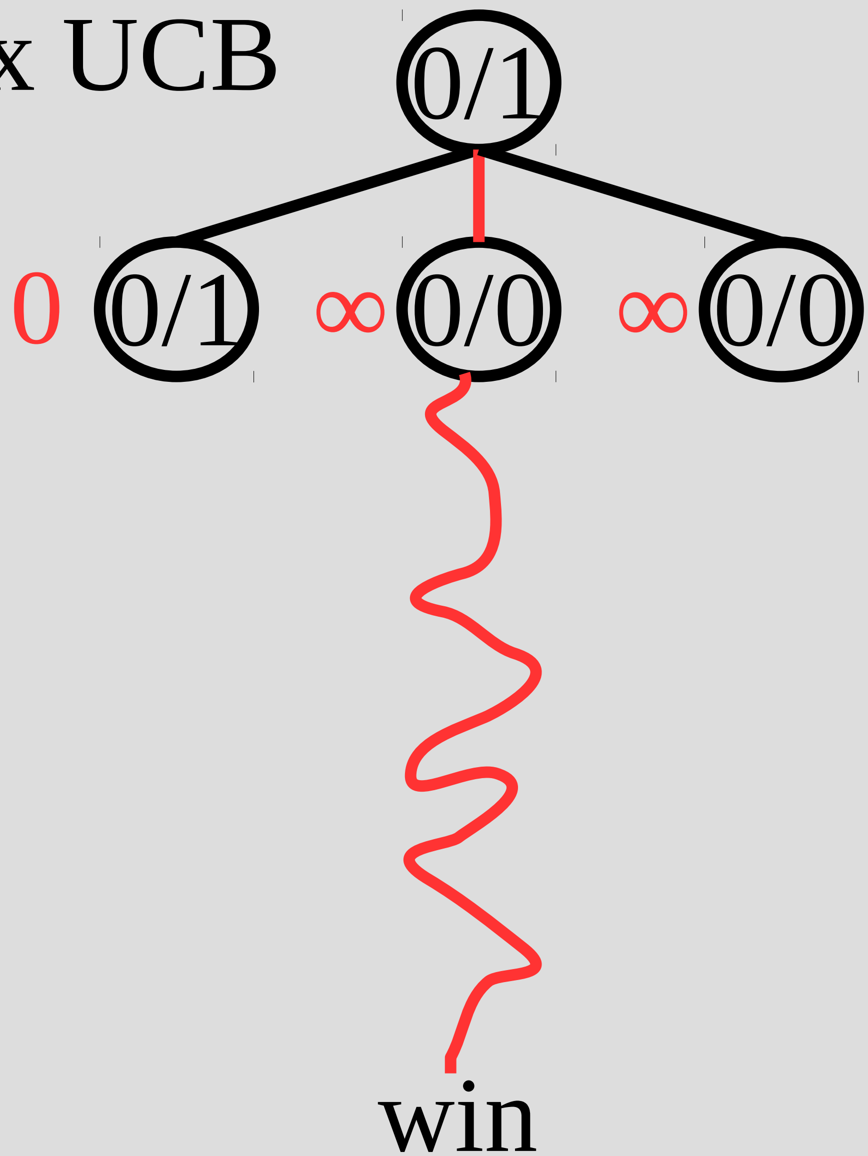
MCTS



update UCB values (all nodes)

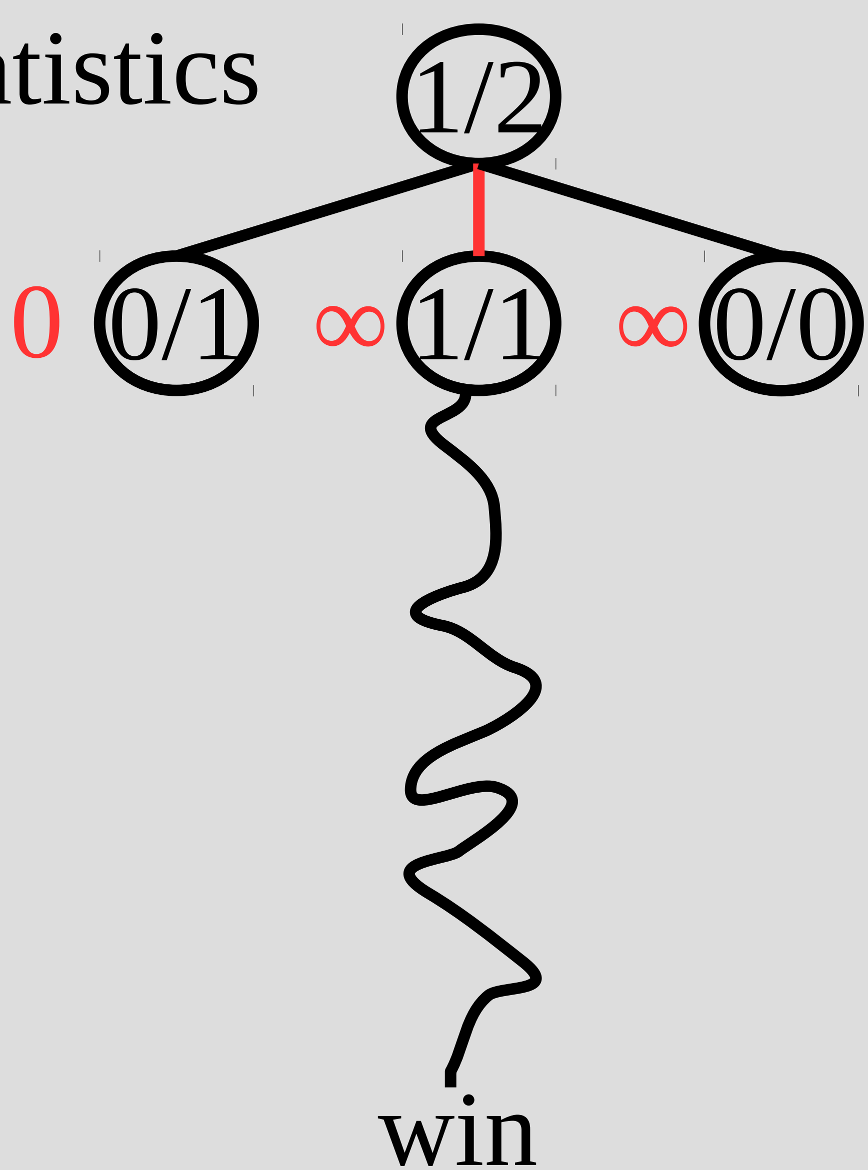
MCTS

select max UCB
& rollout



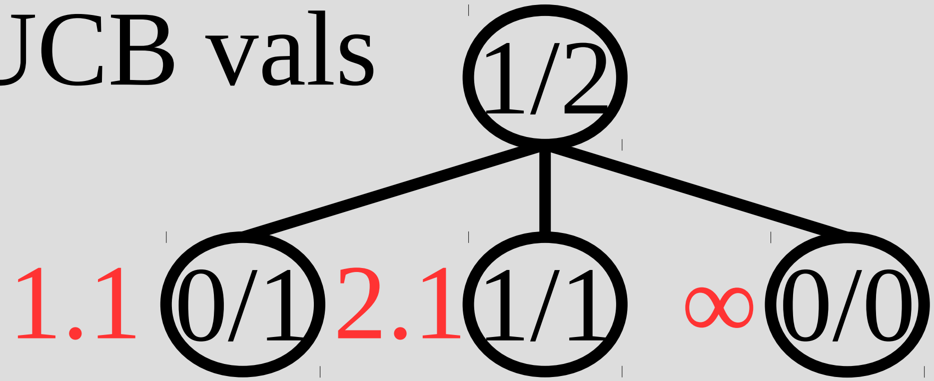
MCTS

update statistics



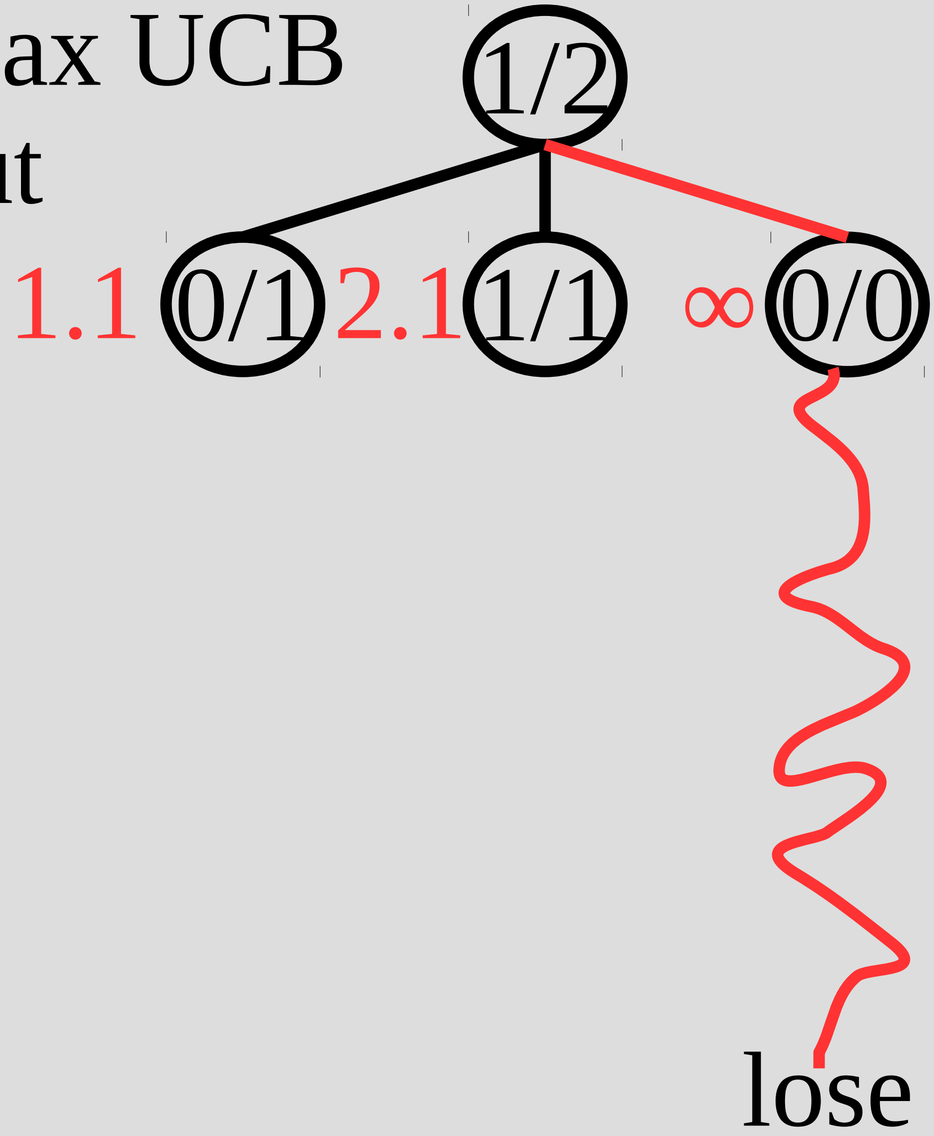
MCTS

update UCB vals



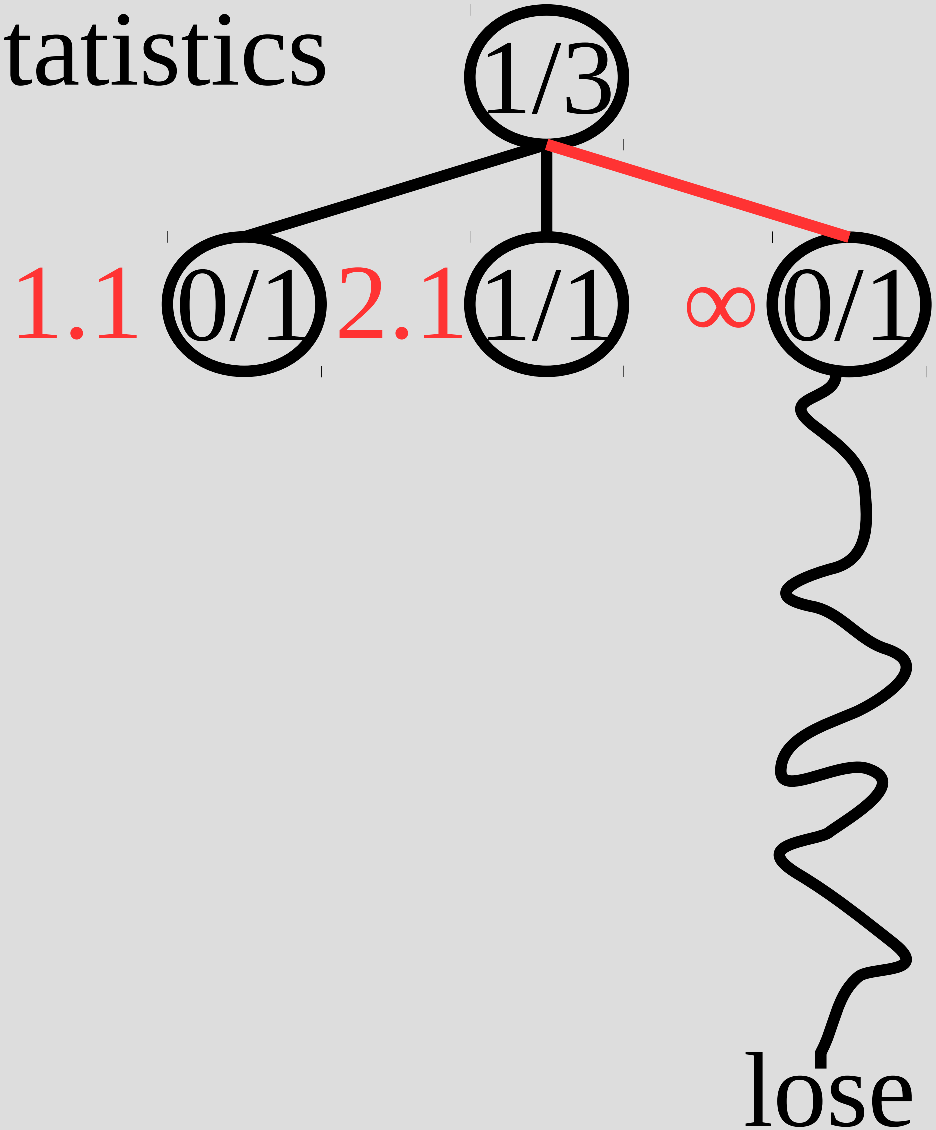
MCTS

select max UCB
& rollout



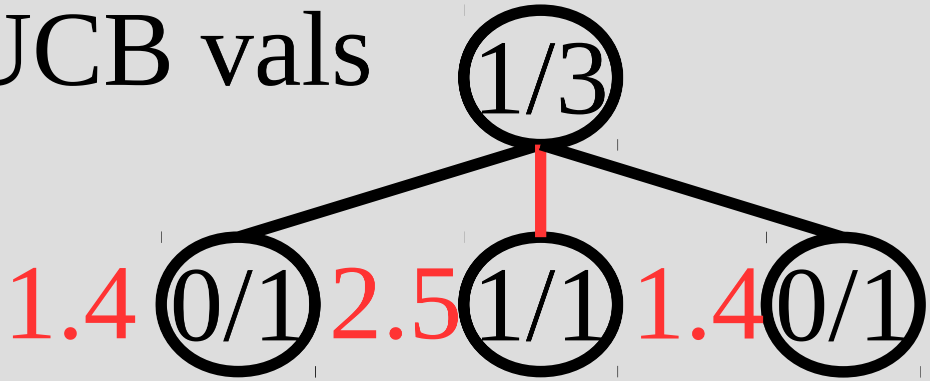
MCTS

update statistics



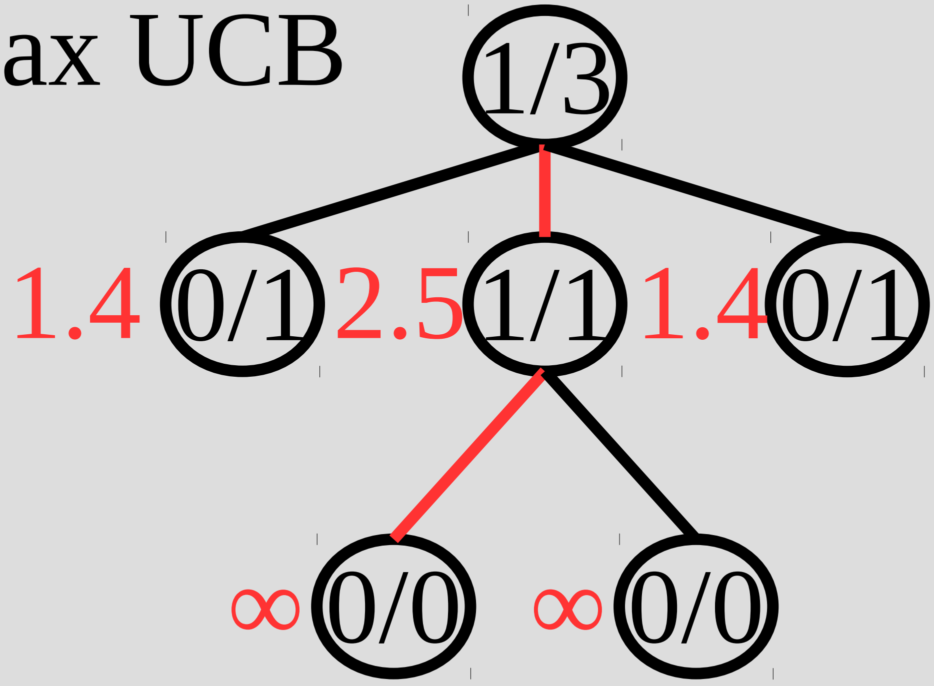
MCTS

update UCB vals



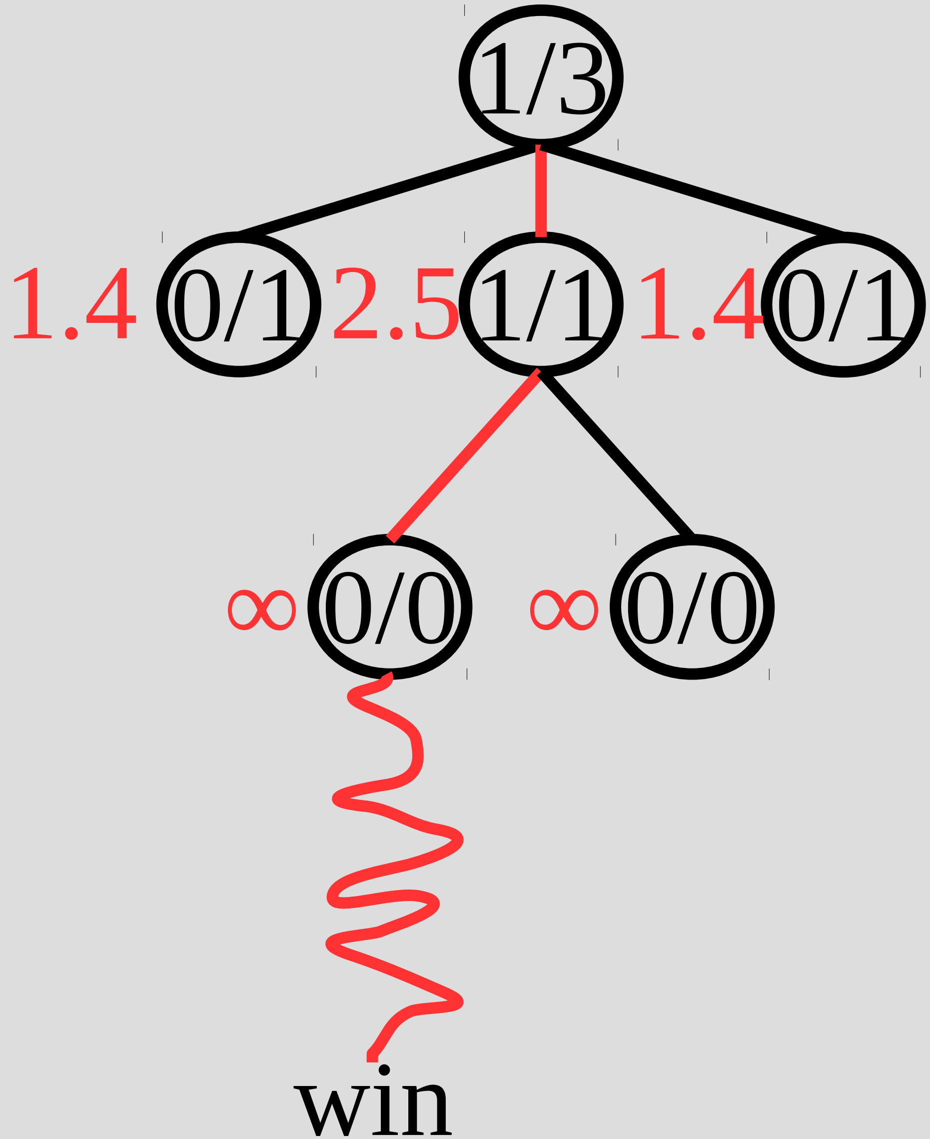
MCTS

select max UCB



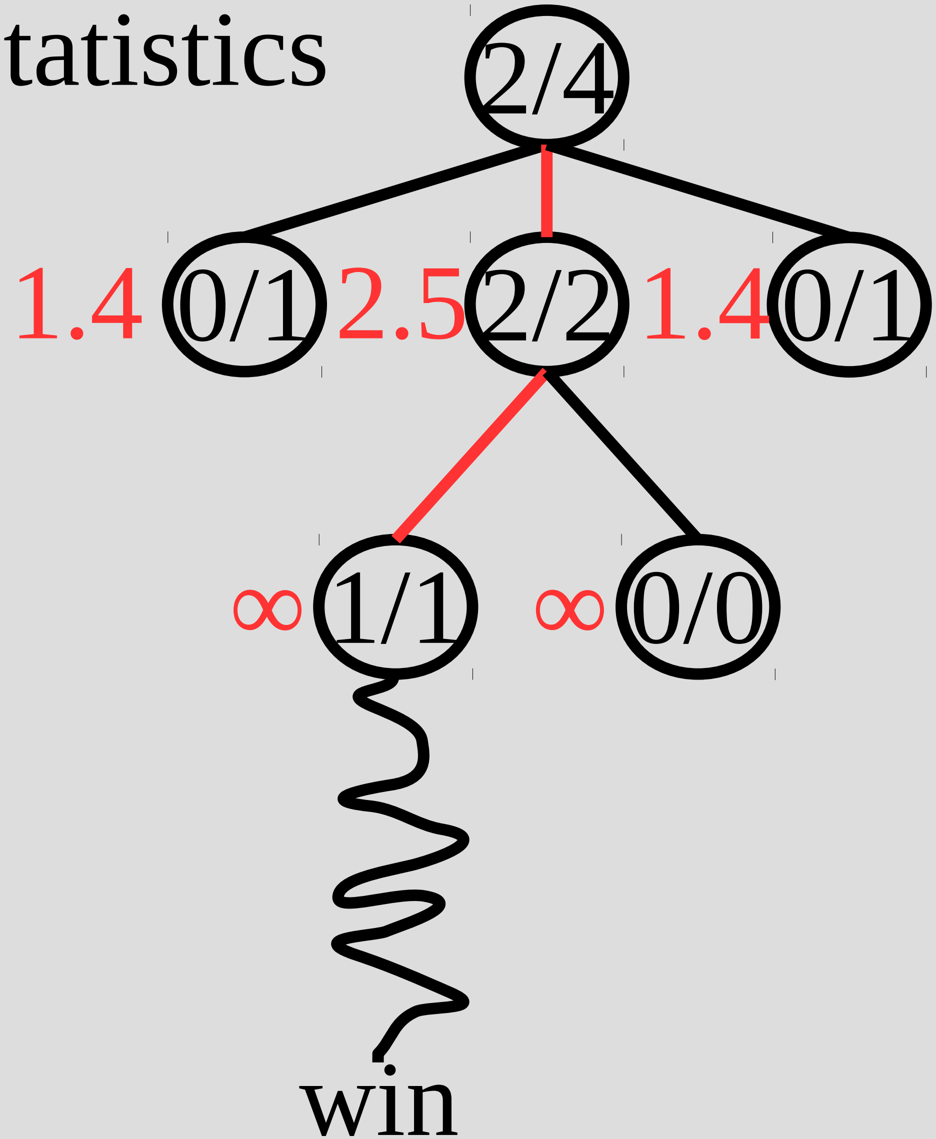
MCTS

rollout



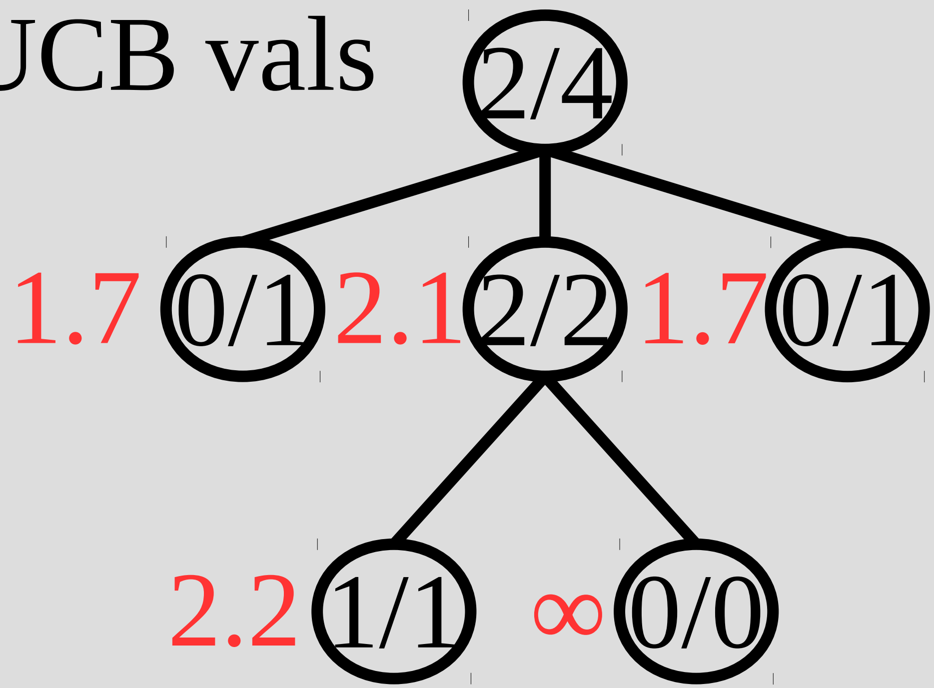
MCTS

update statistics



MCTS

update UCB vals



MCTS

Pros:

(1) The “random playouts” are essentially generating a mid-state evaluation for you

(2) Has shown to work well on wide & deep trees, can also combine distributed comp.

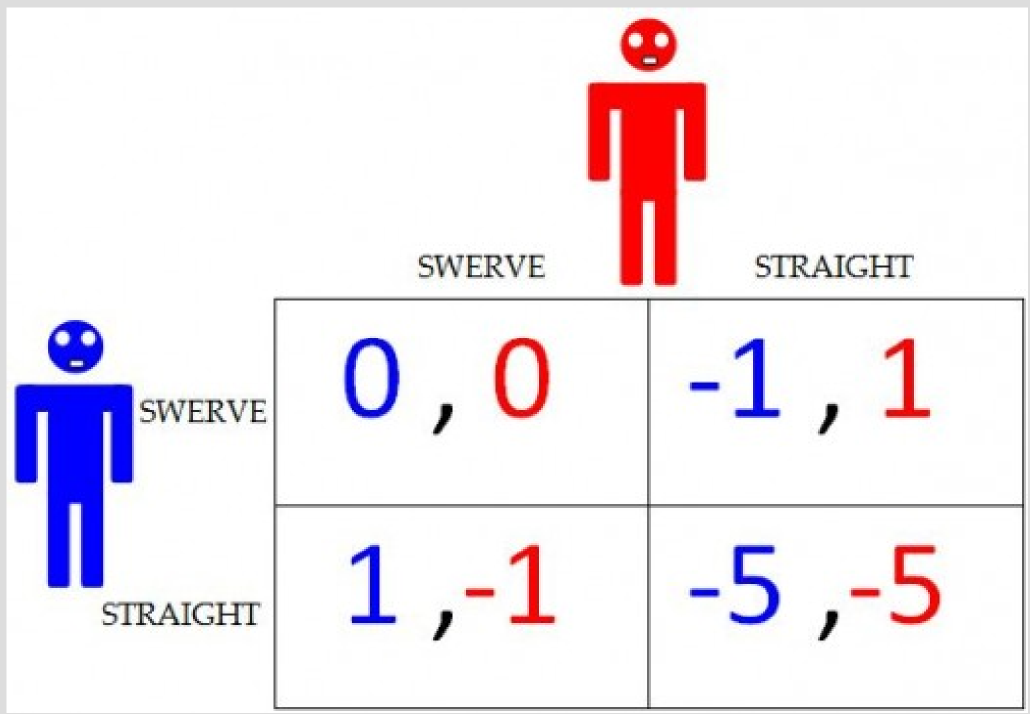
Cons:

(1) Does not work well if the state does not “build up” well

(2) Often does not work on 1-player games

Game theory

Typically game theory uses a <u>payoff matrix</u> to represent the value of actions



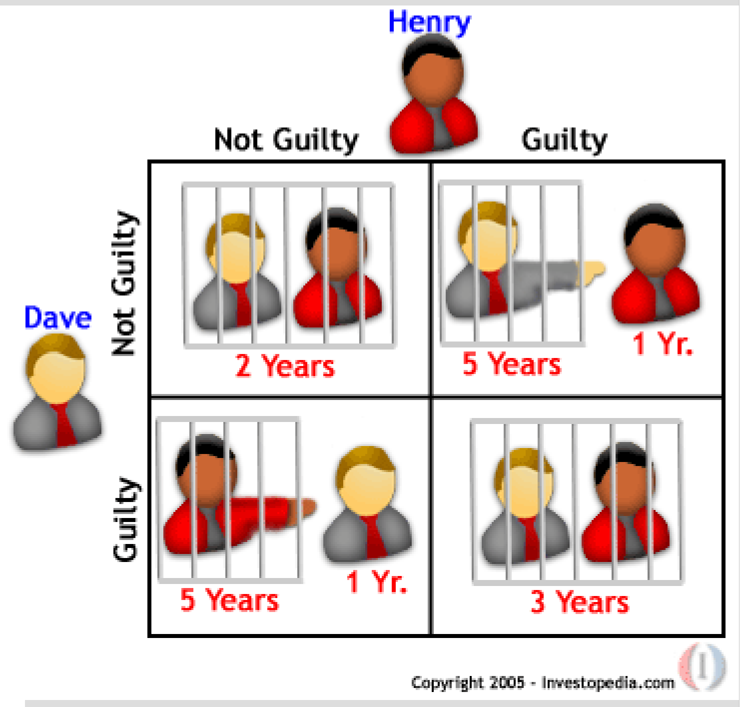
The first value is the reward for the left player, right for top (positive is good for both)

Dominance & equilibrium

Here is the famous “prisoner's dilemma”

Each player chooses one action without knowing the other's and the is only played once

		PRISONER 2	
		Confess	Lie
PRISONER 1	Confess	-8 , -8	0 , -10
	Lie	-10 , 0	-1 , -1

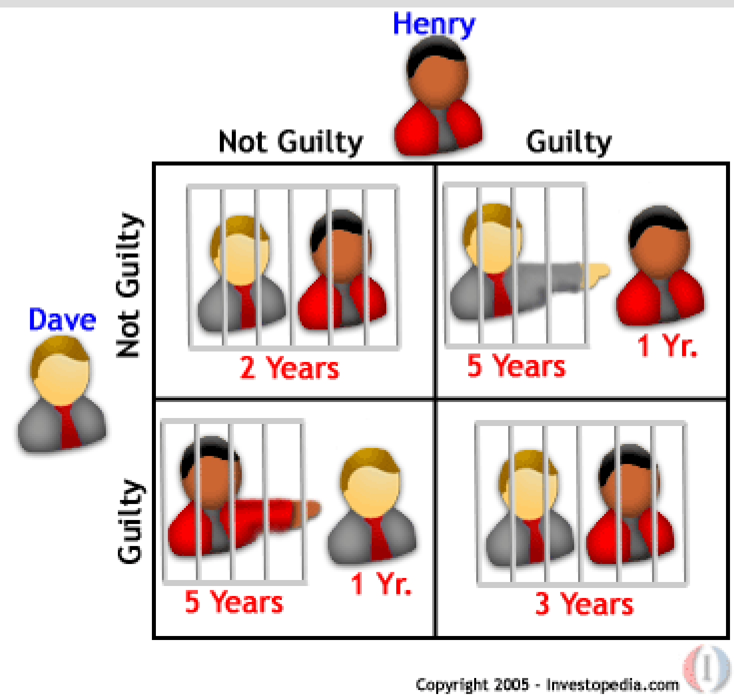


Dominance & equilibrium

What option would you pick?

Why?

		PRISONER 2	
		Confess	Lie
PRISONER 1	Confess	-8 , -8	0 , -10
	Lie	-10 , 0	-1 , -1



Dominance & equilibrium

What would a rational agent pick?

If prisoner 2 confesses, we are in the first column... -8 if we confess, or -10 if we lie
--> Thus we should confess

If prisoner 2 lies, we are in the second column,
0 if we confess,
-1 if we lie
--> We should confess

		PRISONER 2	
		Confess	Lie
PRISONER 1	Confess	-8 , -8	0 , -10
	Lie	-10 , 0	-1 , -1

Dominance & equilibrium

It turns out regardless of the other player's action, it is in our personal interest to confess

This is the <u>Nash equilibrium</u>, as any deviation of our strategy (i.e. lying) can result in a lower score (i.e. if opponent confesses)

The Nash equilibrium looks at the worst case and is greedy

		PRISONER 2	
		Confess	Lie
PRISONER 1	Confess	-8 , -8	0 , -10
	Lie	-10 , 0	-1 , -1

Dominance & equilibrium

Alternatively, a Pareto optimum is a state where no other state is strictly better for all players

If the PD game, [-8, -8] is a Nash equilibrium, but is not a Pareto optimum (as [-1, -1] better for both players)

However [-10,0] is also a Pareto optimum...

		PRISONER 2	
		Confess	Lie
PRISONER 1	Confess	-8 , -8	0 , -10
	Lie	-10 , 0	-1 , -1

Dominance & equilibrium

Every game has at least one Nash equilibrium and Pareto optimum, however...

- Nash equilibrium might not be the best outcome for all players (like PD game, assumes no cooperation)

- A Pareto optimum might not be stable (in PD the [-10,0] is unstable as player 1 wants to switch off "lie" and to "confess" if they play again or know strategy)

Dominance & equilibrium

Find the Nash and Pareto for the following:
(about lecturing in a certain csci class)

	Student: pay attention	Student: sleep
Teacher: prepare well	5, 5	-2, 2
Teacher: slack off	1, -5	0, 0

Find best strategy

How do we formally find a Nash equilibrium?

If it is zero-sum game, can use minimax
as neither player wants to switch for Nash
(our PD example was not zero sum)

Let's play a simple number game: two players
write down either 1 or 0 then show each other.
If the sum is odd, player one wins. Otherwise,
player 2 wins (on even sum)

Find best strategy

This gives the following payoffs:

Player 2

Player 1	Pick 0	Pick 1
Pick 0	-1, 1	1, -1
Pick 1	1, -1	-1, 1

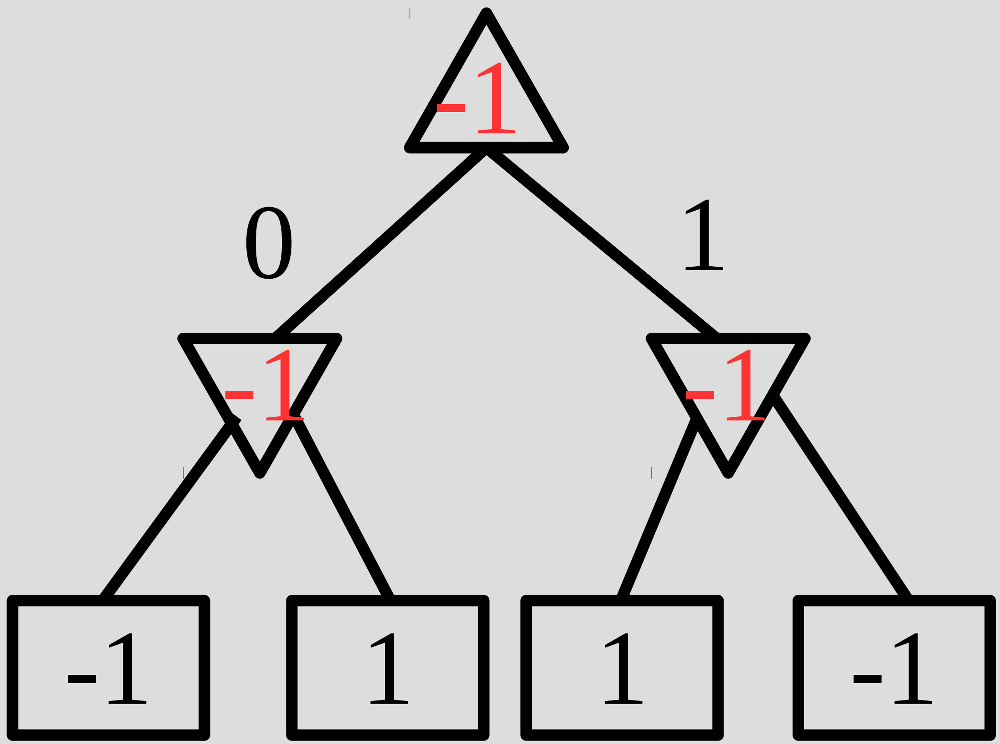
(player 1's value first, then player 2's value)

We will run minimax on this tree twice:

1. Once with player 1 knowing player 2's move (i.e. choosing after them)
2. Once with player 2 knowing player 1's move

Find best strategy

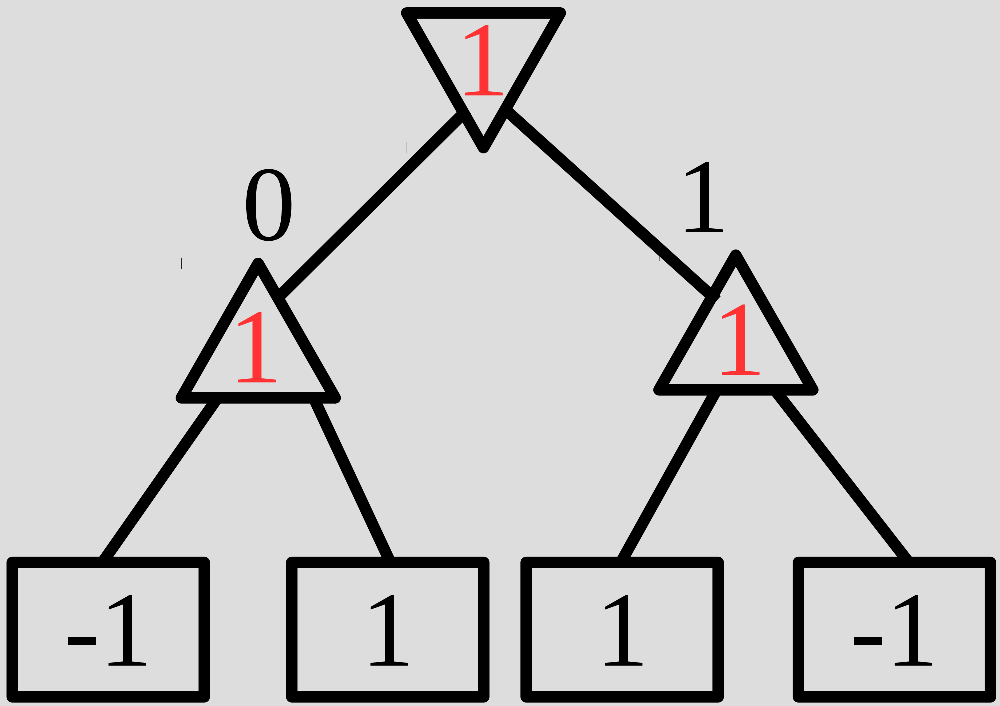
Player 1 to go first (max):



If player 1 goes first, it will always lose

Find best strategy

Player 2 to go first (min):



If player 2 goes first, it will always lose

Find best strategy

This is not useful, and only really tells us that the best strategy is between -1 and 1 (which is fairly obvious)

This minimax strategy can only find pure strategies (i.e. you should play a single move 100% of the time)

To find a mixed strategy, we need to turn to linear programming

Find best strategy

A pure strategy is one where a player always picks the same strategy (deterministic)

A mixed strategy is when a player chooses actions probabilistically from a fixed probability distribution (i.e. the percent of time they pick an action is fixed)

If one strategy is better or equal to all others across all responses, it is a dominant strategy

Find best strategy

The definition of a Nash equilibrium is when your opponent has no incentive to change

So we will only consider our opponent's rewards (and not consider our own)

This is a bit weird since we are not considering our own rewards at all, which is why the Nash equilibrium is sometimes criticized

Find best strategy

First we parameterize this and make the tree stochastic:

Player 1 will choose action “0” with probability p, and action “1” with (1-p)

If player 2 always picks 0, so the payoff for p2:
$(1)p + (-1)(1-p)$
If player 2 always picks 1, so the payoff for p2:
$(-1)p + (1)(1-p)$

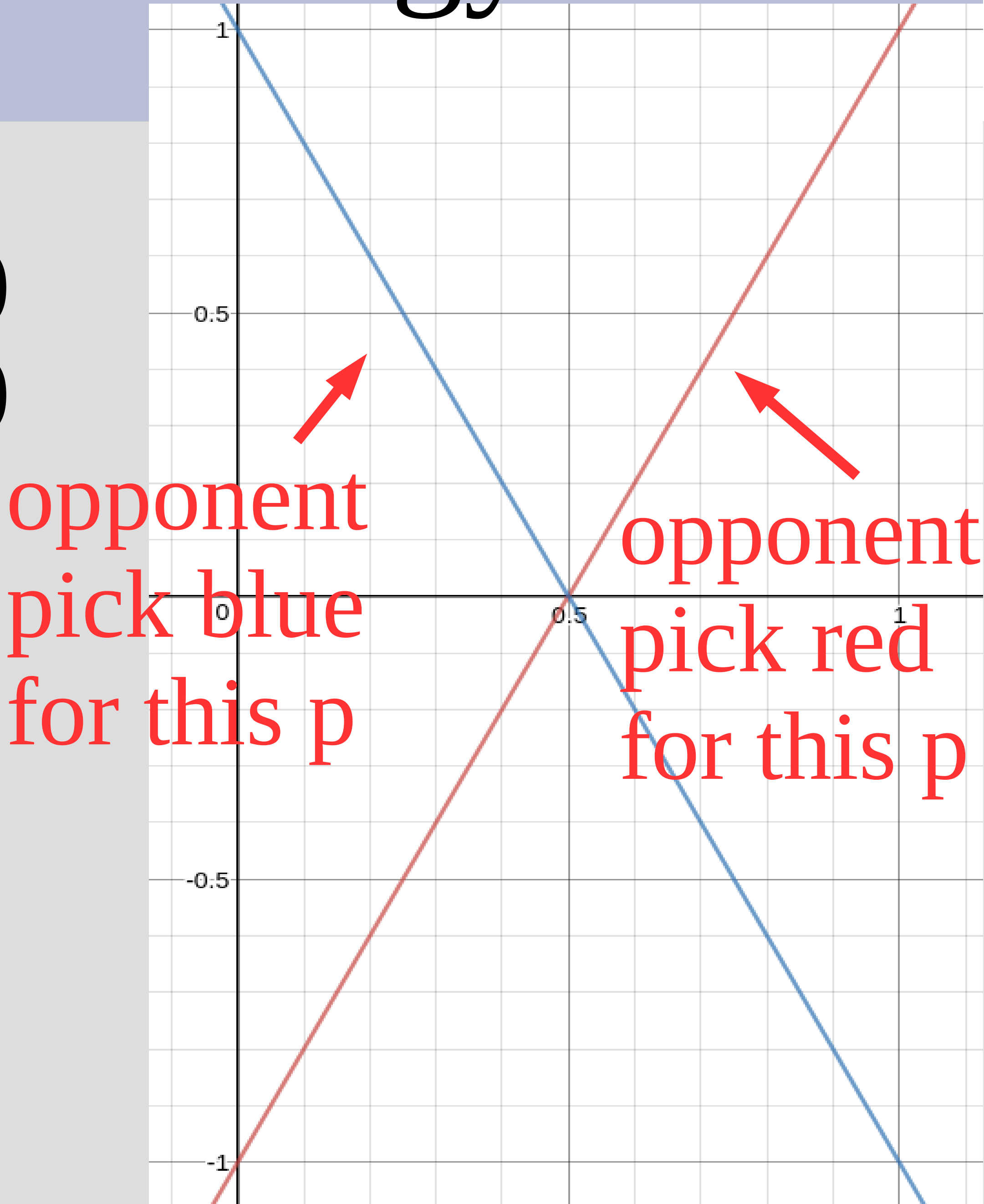
Find best strategy

Plot these two lines:

$$U = (1)p + (-1)(1-p)$$
$$U = (-1)p + (1)(1-p)$$

As we maximize, the opponent gets to pick which line to play

Thus we choose the intersection



Find best strategy

Thus we find that our best strategy is to play 0 half the time and 1 the other half

The result is we win as much as we lose on average, and the overall game result is 0

Player 2 can find their strategy in this method as well, and will get the same 50/50 strategy (this is not always the case that both players play the same for Nash)

Find best strategy

We have two actions, so one parameter (p) and thus we look for the intersections of lines

If we had 3 actions (rock-paper-scissors), we would have 2 parameters and look for the intersection of 3 planes (2D)

This can generalize to any number of actions (but not a lot of fun)

		Player 2		
		Stone	Paper	Scissors
Player 1	Stone	$(0, 0)$	$(-1, 1)$	$(1, -1)$
	Paper	$(1, -1)$	$(0, 0)$	$(-1, 1)$
	Scissors	$(-1, 1)$	$(1, -1)$	$(0, 0)$

Find best strategy

How does this compare on PD?

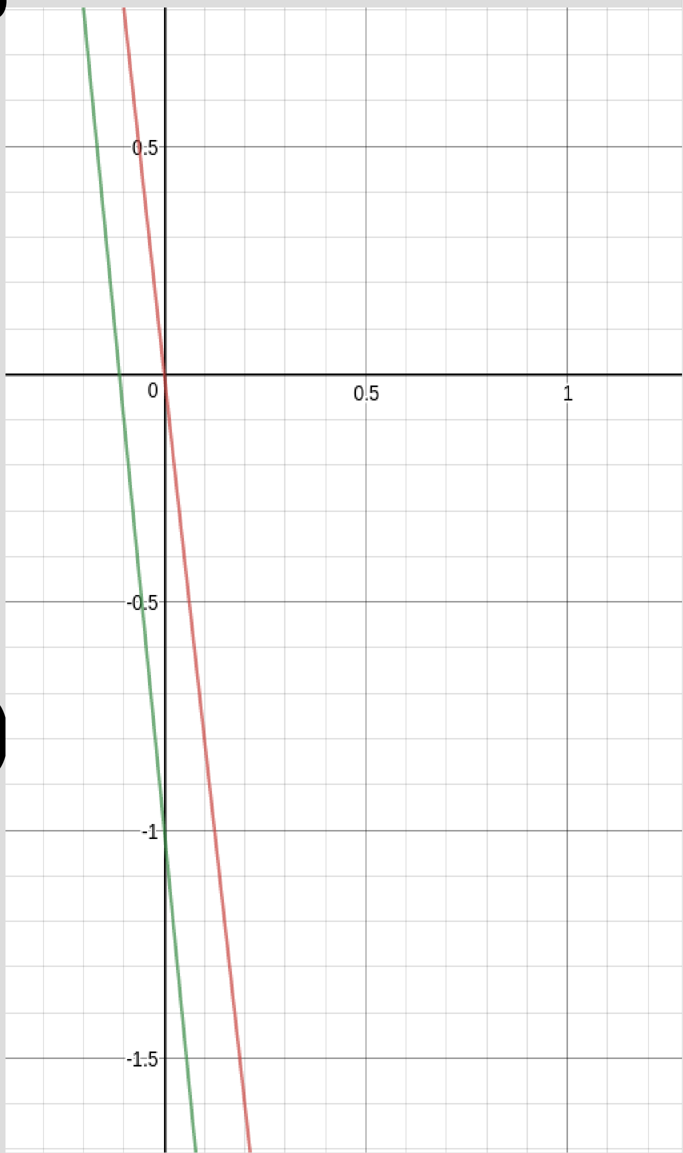
	Confess	Lie
Confess	-8 , -8	0 , -10
Lie	-10 , 0	-1 , -1

Player 1: p = prob confess...

P2 Confesses: $-8*p + 0*(1-p)$

P2 Lies: $-10*p + (-1)*(1-p)$

Cross at negative p, but red line is better (confess)

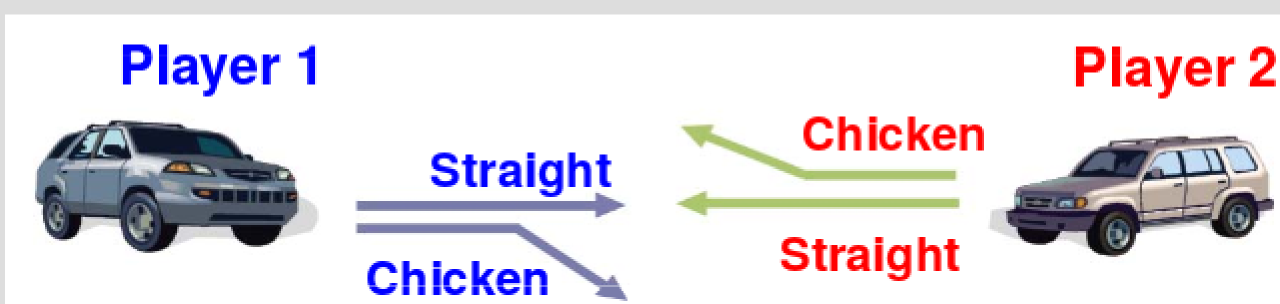


Chicken

What is Nash for this game?
What is Pareto optimum?

	S	C
S	-10, -10	1, -1
C	-1, 1	0, 0

Game of Chicken



Chicken

To find Nash, assume we (blue) play S probability p, C prob 1-p

	S	C
S	-10, -10	1, -1
C	-1, 1	0, 0

Column 1 (red=S): p*(-10) + (1-p)*(1)
Column 2 (red=C): p*(-1) + (1-p)*(0)

Intersection: -11*p + 1 = -p, p = 1/10

Conclusion: should always go straight 1/10 and chicken 9/10 the time

Chicken

	s	c
s	-10, -10	1, -1
c	-1, 1	0, 0

We can see that 10% straight makes the opponent not care what strategy they use:

(Red numbers)

100% straight: (1/10)*(-10) + (9/10)*(1) = -0.1

100% chicken: (1/10)*(-1) + (9/10)*(0) = -0.1

50% straight: (0.5)*[(1/10)*(-10) + (9/10)*(1)]
+ (0.5)*[(1/10)*(-1) + (9/10)*(0)]
=(0.5)*[-0.1] + (0.5)*[-0.1] = -0.1

Chicken

	s	c
s	-10, -10	1, -1
c	-1, 1	0, 0

The opponent does not care about action, but you still do (never considered our values)

Your rewards, opponent 100% straight:

(0.1)*(-10) + (0.9)*(-1) = -1.9

Your rewards, opponent 100% curve:

(0.1)*(1) + (0.9)*(0) = 0.1

The opponent also needs to play at your value intersection to achieve Nash

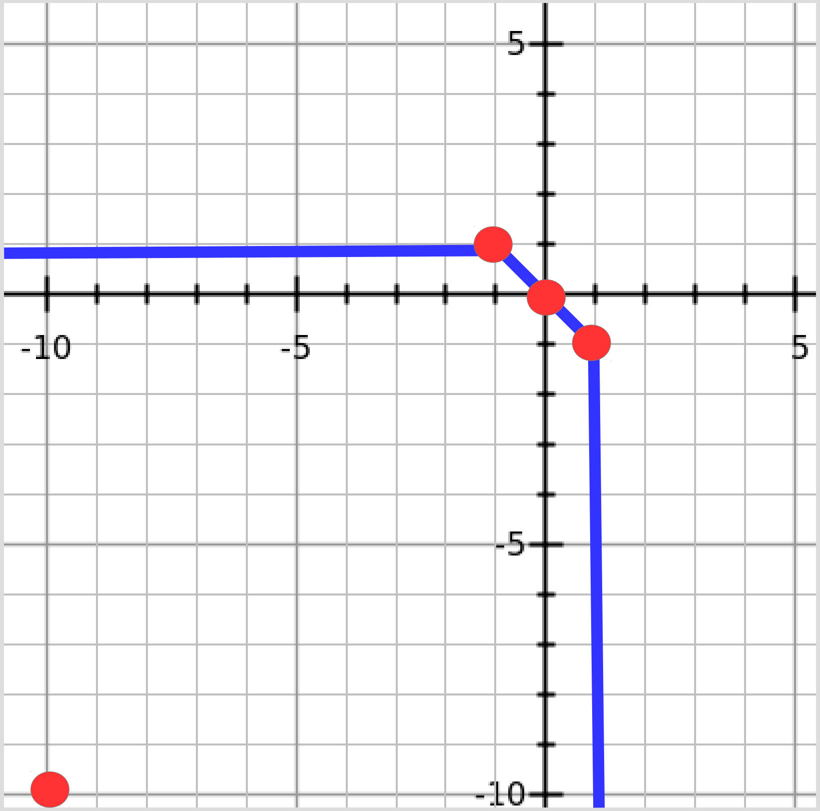
Chicken

Pareto optimum?
All points except (-10,10)

Can think about this
as taking a string from the
top right and bringing the
it down & left

Stop when string going
straight left and down

	S	C
S	-10, -10	1, -1
C	-1, 1	0, 0



Repeated games

In repeated games, things are complicated

For example, in the basic PD, there is no benefit to “lying”

		PRISONER 2	
		Confess	Lie
PRISONER 1	Confess	-8 , -8	0 , -10
	Lie	-10 , 0	-1 , -1

However, if you play this game multiple times, it would be beneficial to try and cooperate and stay in the [lie, lie] strategy

Repeated games

One way to do this is the <u>tit-for-tat</u> strategy:

1. Play a cooperative move first turn
2. Play the type of move the opponent last played every turn after (i.e. answer competitive moves with a competitive one)

This ensure that no strategy can “take advantage” of this and it is able to reach cooperative outcomes

Repeated games

Two “hard” topics (if you are interested) are:

1. We have been talking about how to find best responses, but it is very hard to take advantage if an opponent is playing a sub-optimal strategy

2. How to “learn” or “convince” the opponent to play cooperatively if there is an option that benefits both (yet dominated)