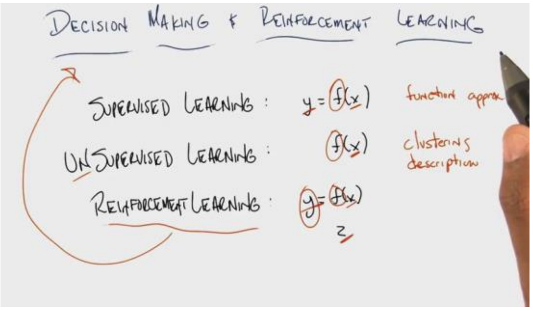
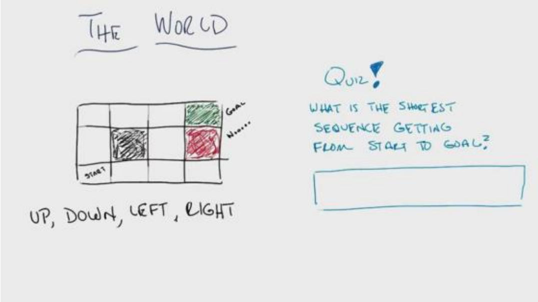
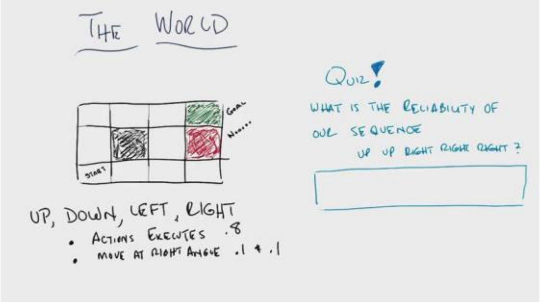
RL 1 - Markov Decision Process

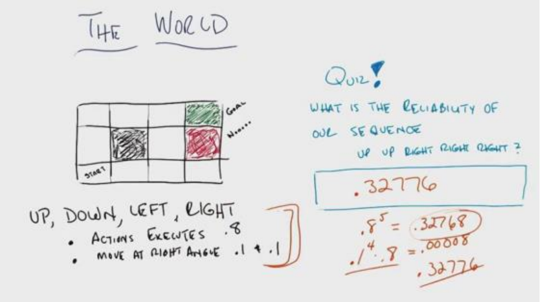

DECISION MAKING & REINFORCEMENT LEARNING
SUPERVISED LEARNING : y = f(x)
function approx
UNSUPERVISED LEARNING : f(x)
clustering description
REINFORCEMENT LEARNING : y = f(x)
z

- For reinforced learning, given x and z, and try to find function f that generates y

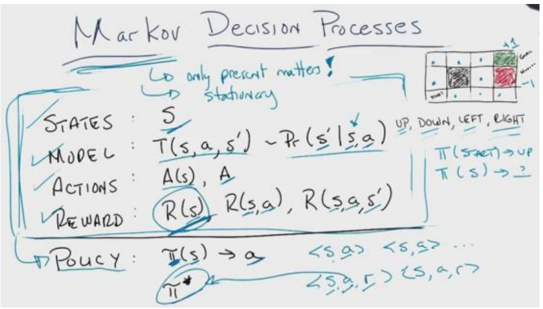


- U, U, R, R, R or R, R, U, U, R
- Usually multiple solutions available


THE WORLD
GOAL
START
UP, DOWN, LEFT, RIGHT
• ACTIONS EXECUTES .8
• MOVE AT RIGHT ANGLE .1 + .1
QUIZ!
WHAT IS THE RELIABILITY OF OUR SEQUENCE
UP UP RIGHT RIGHT RIGHT ?



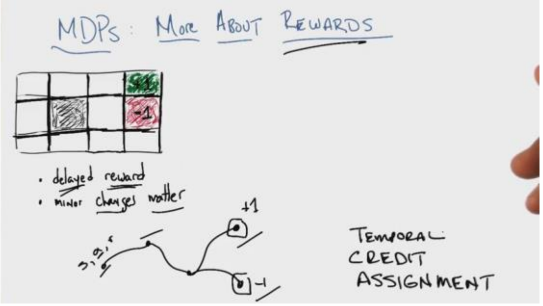
- Stochasticity of movements
- Need to consider the path to the right
- Need to finish in green spot after 5 steps, no more steps available

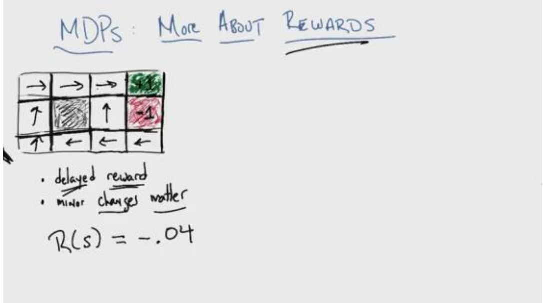


- add historical states to make the process Markov
- RL: given <state,action,reward> pairs as training samples, and try to find the optimal policy pi or

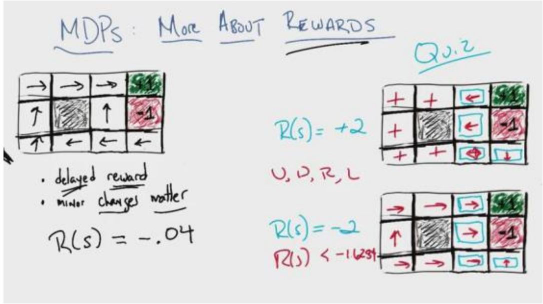
optimal solution

- SL: given <state,action> pairs as training samples, and try to find the optimal polity pi
- Assumptions: you know which state S, and the reward for taking actions
- Policy does not tell you the sequency of actions (i.e. plan) to reach success, but what actions to take at particular state
- The problem: given the MDP, how to find the optimal policy pi?

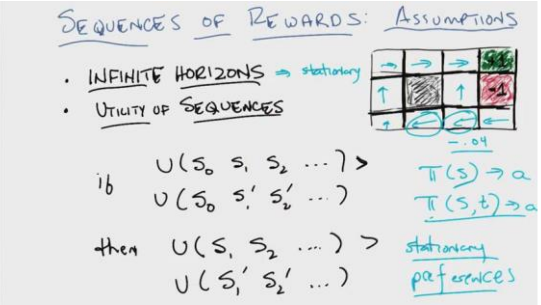




- MDPs: more about rewards
- Delayed rewards
- Temporal credit assignment, where credit = 1 or -1
- reward R(s) = -0.04 for each state except the terminating positions
- The arrow the global optimal solution (or plan)
- The reward function may change the global solution greatly: totally different path given different reward functions



- Big negative rewards to end game as soon as possible
- Big positive rewards to continue game as long as possible
- Medium negative rewards: in the game to get the +1 reward
- Big positive rewards: just in the game
- Big negative rewards: get out of game ASAP
- Reward function is domain knowledge: if you want to use MDP to describe a world, you need to think carefully how you set up reward functions



SEQUENCES OF REWARDS: ASSUMPTIONS

- INFINITE HORIZONS ⇒ stationary
- UTILITY OF SEQUENCES

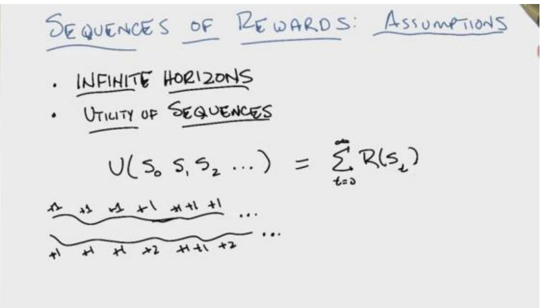
if $U(s_0\ s_1\ s_2 \dots) >$
$U(s_0\ s_1'\ s_2' \dots)$

then $U(s_1\ s_2 \dots) >$
$U(s_1'\ s_2' \dots)$

$\pi(s) \to a$

$\pi(s,t) \to a$

stationary preferences



SEQUENCES OF REWARDS: ASSUMPTIONS

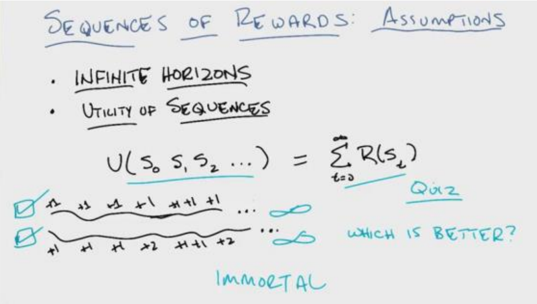
- INFINITE HORIZONS
- UTILITY OF SEQUENCES

$U(s_0\ s_1\ s_2 \dots) = \sum_{t=0}^{\infty} R(s_t)$

+1 +1 +1 +1 +1 +1 +1 ...

+1 +1 +1 +2 +1 +1 +2 ...

- Infinite horizon: you can live for ever to try, till get the terminating states => need to consider reward function, as well as how much time left (i.e. horizon)
- If the horizon is not infinite, for example, game ended within 3 steps when in state (3,1), then probably going UP is the right choice, instead of going left
- In the infinite horizon, the process is stationary, where the action only depends on the state. Or, as long as I am at this particular state, I will always take the same action. Policy pi(S) -> a
- In the finite horizon, the process is non-stationary, where the action may be different depending on how much time left at the same state. Policy pi(S,t) -> a
- Utility of sequence (vs utility of a single step): U(s_0,s_1,…,)
- Lemma (stationarity of preference): if U(S0,S1,S2,…)>U(S0,S1',S2',…) (utility of sequence starting at the same state s0), then
- U(S1,S2,…) > U(S1',S2',…)
- Or, if I prefer a sequence over another sequence today, then I prefer the same sequence tomorrow
- Note: the utility of a sequence = sum of rewards along the path, then the Lemma is true. On the other hand, if Lemma is true for any sequence, then the utility of sequence = sum of rewards along the path.



SEQUENCES OF REWARDS: ASSUMPTIONS

- INFINITE HORIZONS
- UTILITY OF SEQUENCES

$U(s_0\ s_1\ s_2 \dots) = \sum_{t=0}^{\infty} R(s_t)$

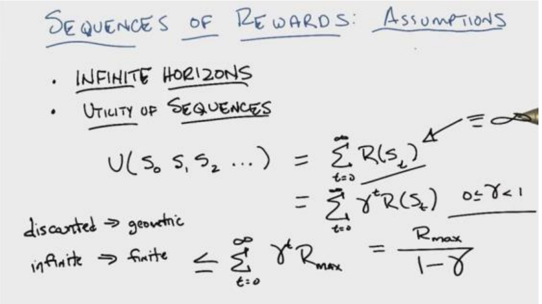
Quiz

☑ +1 +1 +1 +1 +1 +1 +1 ... ∞

☑ +1 +1 +1 +2 +1 +1 +2 ... ∞

WHICH IS BETTER?

IMMORTAL



SEQUENCES OF REWARDS: ASSUMPTIONS

- INFINITE HORIZONS
- UTILITY OF SEQUENCES

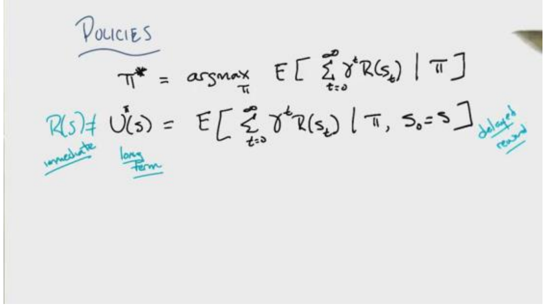
$U(s_0\ s_1\ s_2 \dots) = \sum_{t=0}^{\infty} R(s_t) = \infty$

$= \sum_{t=0}^{\infty} \gamma^t R(s_t) \quad 0 \le \gamma < 1$

discounted ⇒ geometric

infinite ⇒ finite $\le \sum_{t=0}^{\infty} \gamma^t R_{max} = \frac{R_{max}}{1-\gamma}$

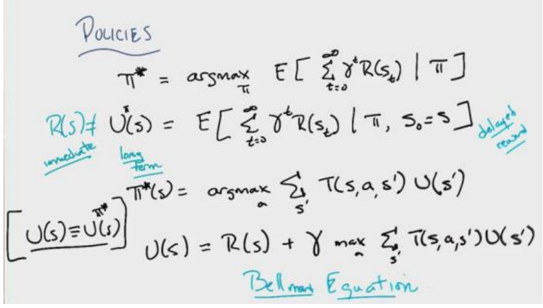
- Discounted rewards by adding the factor gamma^t
- Infinite horizon but finite total reward



Policies

$\pi^* = \arg\max_{\pi} E[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi]$

$R(s) \neq U^{\pi}(s) = E[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi, s_0 = s]$



Policies

$\pi^* = \arg\max_{\pi} E[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi]$

$R(s) \neq U^{\pi}(s) = E[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi, s_0 = s]$

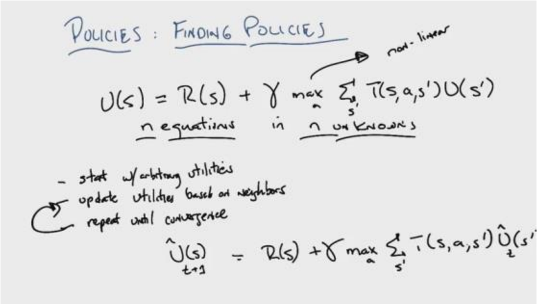
$\pi^*(s) = \arg\max_a \sum_{s'} T(s,a,s') U(s')$

$\boxed{U(s) \equiv U^{\pi^*}(s)}$

$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s') U(s')$

Bellman Equation

- The optimal policy pi, will optimize the total expected discount rewards
- The utility given policy pi is the expected total discount rewards given policy pi and starting state S
- T(S,a,S') is the transition probability that we start at state S and end up at state S' under activation a
- U(S) is the total discount rewards given optimal policy
- Bellman equation is the key to solve MDP problem

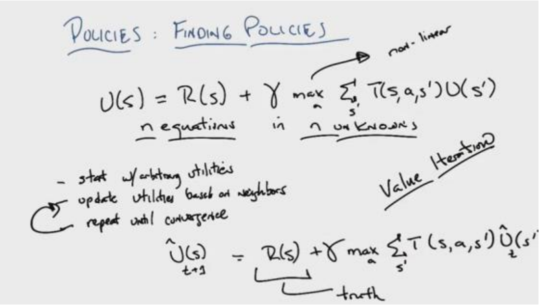


Policies : Finding Policies

$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s') U(s')$

- start w/ arbitrary utilities
- update utilities based on neighbors
- repeat until convergence

$\hat{U}_{t+1}(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s') \hat{U}_t(s')$



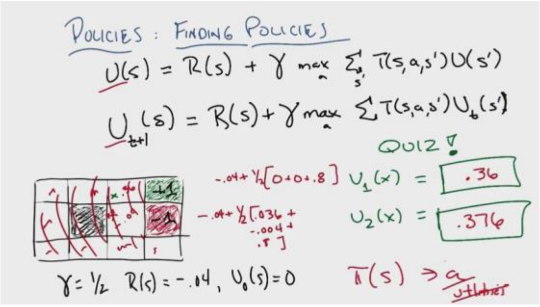
Policies : Finding Policies

$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s') U(s')$

- start w/ arbitrary utilities
- update utilities based on neighbors
- repeat until convergence

$\hat{U}_{t+1}(s) = \underbrace{R(s)}_{\text{truth}} + \gamma \max_a \sum_{s'} T(s,a,s') \hat{U}_t(s')$

- For Bellman equation, there are N equations with N unknowns, but max function make it not-linear => algorithm is used to solve these equations (called value iterations)
- U_t(S) is arbitrary values, while R(S), the instant reward function, is the truth
- Adding more truth through loop will converge to the solution
- Value iterations: does not give you the right answer, but closer and closer answers
- Once the utility function is solved, then it's easy to solve the optimal policy pi



Policies : Finding Policies

$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s') U(s')$

$U_{t+1}(s) = R(s) + \gamma \max_a \sum T(s,a,s') U_t(s')$

Quiz!

$-.04 + \frac{1}{2}[0 + 0 + .8]$ $U_1(x) = \boxed{.36}$

$-.04 + \frac{1}{2}[.036 + -.004 + .8]$ $U_2(x) = \boxed{.376}$

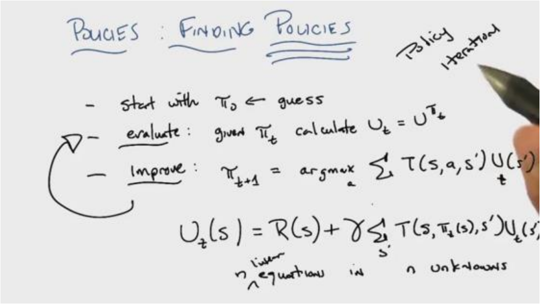
$\gamma = \frac{1}{2}$ $R(s) = -.04$, $U_0(s) = 0$

$\pi(s) \rightarrow \underline{a}$ utilities

- For state X, better to go to green spot with probability 0.8, while going to other two directions with probability 0.1
- For t=2, we need to calculate the U(Y) at t=1 where Y is the grid below X: the optimal policy is to go to the wall, and back to the original state. However, at some point, when the utility of state X is big enough, the optimal policy would be go to state X, instead of wall
- The value iteration works if eventually the value pops out its neighbors => +1 and -1 will pop out to

states over time

- Given utility, we can find policies; but given policies, it's hard to find utility



Policies : Finding Policies

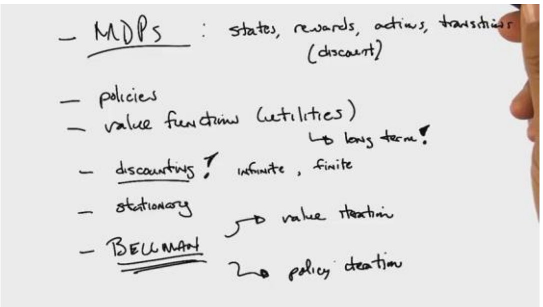
Policy Iteration

- start with $\pi_0 \leftarrow$ guess
- evaluate : given π_t calculate $U_t = U^{\pi_t}$
- improve : $\pi_{t+1} = \arg\max_a \sum T(s,a,s') U_t(s')$

$U_t(s) = R(s) + \gamma \sum_{s'} T(s,\pi_t(s),s') U_t(s'$

n linear equations in n unknowns

- Given policy, we can remove the max function, and turn the Bellman equation into linear equation, and solve it
- Called policy iteration
- Guaranteed to converge: finite number of policies, and you are always getting better



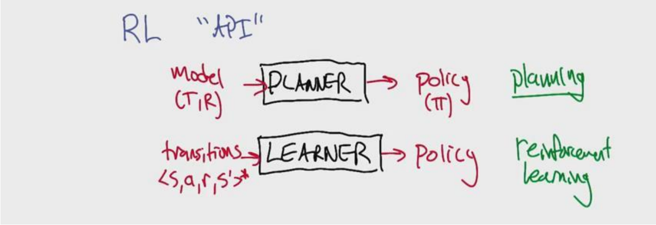
- MDPs : states, rewards, actions, transitions (discount)
- policies
- value functions (utilities) → long term!
- discounting! infinite , finite
- stationary
- BELLMAN → value iteration → policy iteration

- In RL, you don't necessarily know the rewards and transitions

RL 2 - Reinforcement Learning

REINFORCEMENT LEARNING

in Markov Decision Processes...

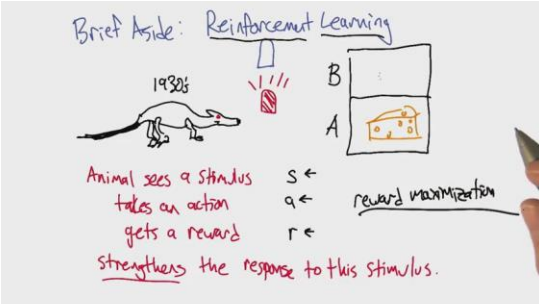


RL "API"

model (T,R) → PLANNER → Policy (π) planning

transitions <s,a,r,s'>* → LEARNER → Policy reinforcement learning

- T: transition matrix; R: reward functions
- \pi: S-> actions; Transitions: samples

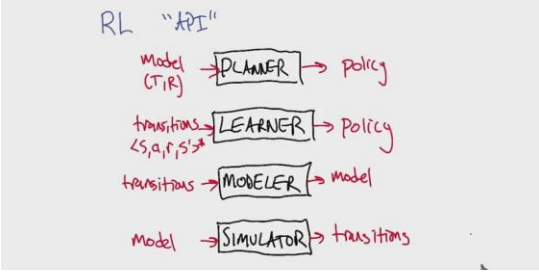


Brief Aside: Reinforcement Learning

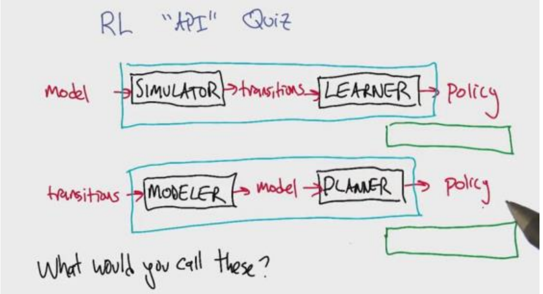
Animal sees a stimulus s ←
takes an action a ←
gets a reward r ←
strengthens the response to this stimulus.

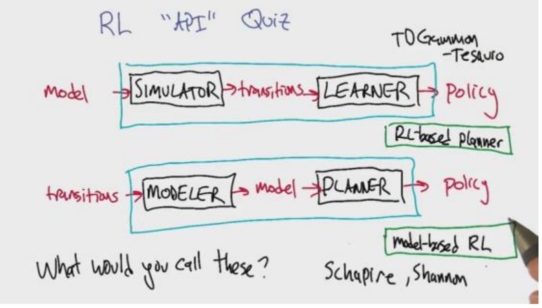
reward maximization

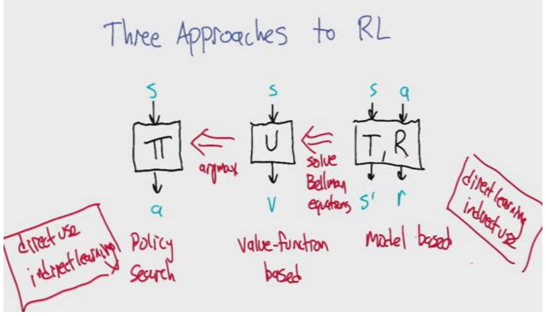
- Stimulus: like seeing the red light <- state; Action: like picking box A <- action
- Reward: like cheese <- reward; Strengthen: means next time sees red light, more likely to go to box A => reinforcing the connection
- How to choose actions to maximize reward as a function of state => reward maximization



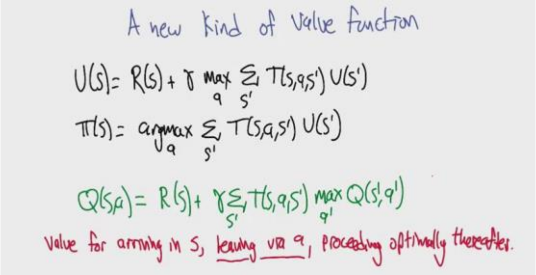
- Modeling: like machine learning problem
- Simulator: simulate samples, time consuming
- One way to do learning: to do modeling and simulating, so that I know a model and what the reinforcement function was, and just do planning







- 3 outputs of RL algorithms: policy search algorithm; value-function based algorithm; model-based RL learner
- From utility to policy: use argmax operation
- From model to utility: value iteration to solve Bellman equation
- Model based RL learning: direct learning, not-direct usage
- We will focus on the second: value-function based learning: medium direct learning and usage



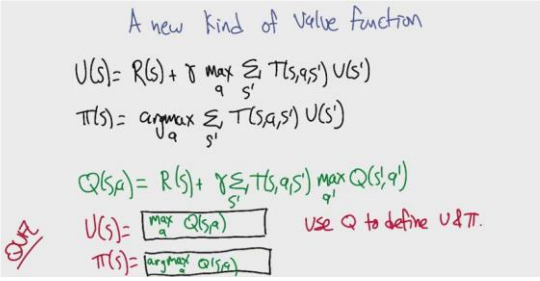
A new kind of Value function

$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s,a,s')\, U(s')$

$\pi(s) = \arg\max_{a} \sum_{s'} T(s,a,s')\, U(s')$

$Q(s,a) = R(s) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q(s',a')$

Value for arriving in s, leaving via a, proceeding optimally thereafter.



A new kind of Value function

$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s,a,s')\, U(s')$

$\pi(s) = \arg\max_{a} \sum_{s'} T(s,a,s')\, U(s')$

$Q(s,a) = R(s) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q(s',a')$

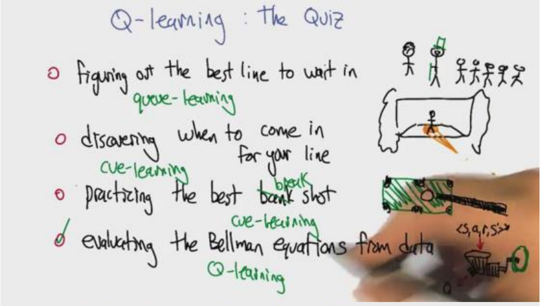
Quiz

$U(s) = \boxed{\max_{a} Q(s,a)}$

$\pi(s) = \boxed{\arg\max_{a} Q(s,a)}$

Use Q to define U & π.

- Q(S,a) is the reward taking a specific action => we can take Q as U, if we always take the best action. The best action is the one maximize the utility (value you get from that point on)



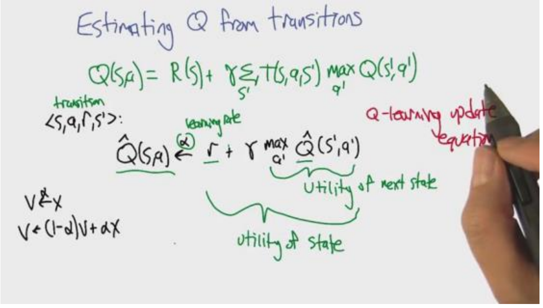
Q-learning : the Quiz

○ figuring out the best line to wait in — queue-learning

○ discovering when to come in for your line — cue-learning

○ practicing the best ~~bank~~ break shot — cue-learning

✓ evaluating the Bellman equations from data — Q-learning



Estimating Q from transitions

$Q(s,a) = R(s) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q(s',a')$

transition $\langle s,a,r,s' \rangle$:

$\underline{\hat{Q}(s,a)} \overset{\alpha}{\leftarrow} \underbrace{\underline{r} + \gamma \underbrace{\max_{a'} \hat{Q}(s',a')}_{\text{utility of next state}}}_{\text{utility of state}}$

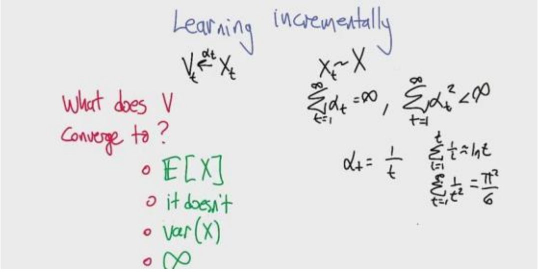
learning rate

Q-learning update equation

$V \overset{\alpha}{\leftarrow} X$

$V \leftarrow (1-\alpha)V + \alpha X$

- We don't have access to R(S) and T(S,a,S'). if known, we can use value iteration and policy iteration to solve Bellman equation for Q(S,a)
- <S, a, r, S'>: we are in state S, take the action a with reward r. The new state is S'
- V<-alpha X: current state is V, and move alpha toward state X
- V<- (1-alpha)V + alpha X



Learning incrementally

$V_t \overset{\alpha_t}{\leftarrow} X_t$

$X_t \sim X$

$\sum_{t=1}^{\infty} \alpha_t = \infty, \quad \sum_{t=1}^{\infty} \alpha_t^2 < \infty$

What does V converge to?

○ E[X]

○ it doesn't

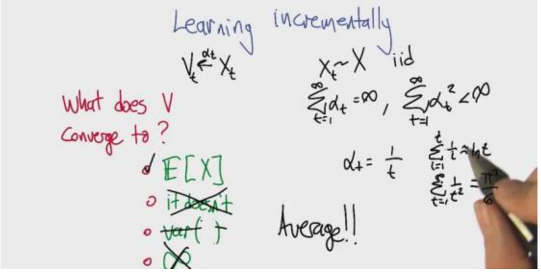
○ var(X)

○ ∞

$\alpha_t = \frac{1}{t}$

$\sum_{t=1}^{T} \frac{1}{t} \approx \ln T$

$\sum_{t=1}^{\infty} \frac{1}{t^2} = \frac{\pi^2}{6}$



Learning incrementally

$V_t \overset{\alpha_t}{\leftarrow} X_t$

$X_t \sim X$ iid

$\sum_{t=1}^{\infty} \alpha_t = \infty, \quad \sum_{t=1}^{\infty} \alpha_t^2 < \infty$

What does V converge to?

✓ E[X]

○ ~~it doesn't~~

○ ~~var( )~~

○ ~~∞~~

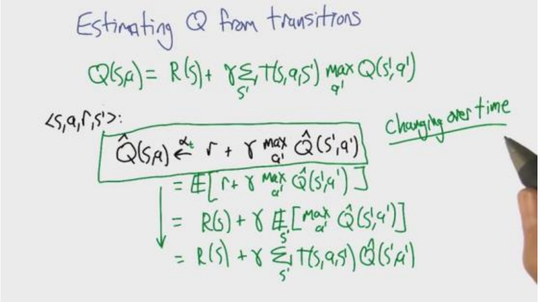
$\alpha_t = \frac{1}{t}$

$\sum_{t=1}^{T} \frac{1}{t} \approx \ln T$

$\sum_{t=1}^{\infty} \frac{1}{t^2} = \frac{\pi^2}{6}$

Average!!

- The X_t samples from the same distribution X, IID
- The learning rate alpha_t satisfies the following conditions: sum is infinite but sum of square is finite



Estimating Q from transitions

$Q(s,a) = R(s) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q(s',a')$

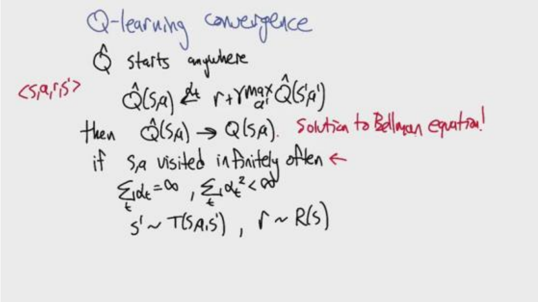
$\langle s,a,r,s' \rangle$:

$\boxed{\hat{Q}(s,a) \xleftarrow{\alpha_t} r + \gamma \max_{a'} \hat{Q}(s',a')}$ Changing over time

$= E[r + \gamma \max_{a'} \hat{Q}(s',a')]$

$= R(s) + \gamma E_{s'}[\max_{a'} \hat{Q}(s',a')]$

$= R(s) + \gamma \sum_{s'} T(s,a,s') \hat{Q}(s',a')$



Q-learning convergence

$\hat{Q}$ starts anywhere

$\langle s,a,r,s' \rangle$ $\hat{Q}(s,a) \xleftarrow{\alpha_t} r + \gamma \max_{a'} \hat{Q}(s',a')$

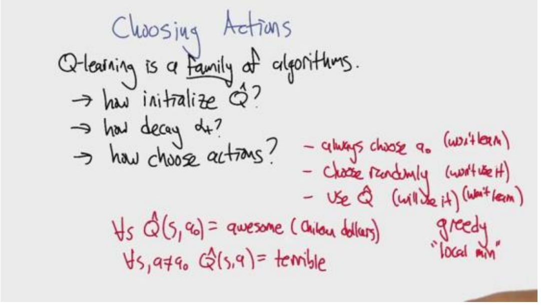
then $\hat{Q}(s,a) \rightarrow Q(s,a)$. Solution to Bellman equation!

if s,a visited infinitely often ←

$\sum_t \alpha_t = \infty$, $\sum_t \alpha_t^2 < \infty$

$s' \sim T(s,a,s')$, $r \sim R(s)$

- Q learning:
- Under alpha learning, the right side will go to the expectation over time
- The green formula derivation is not quite right, as Q hat is changing over time
- In order to be convergent, need to visit states for infinite times



Choosing Actions

Q-learning is a family of algorithms.

→ how initialize $\hat{Q}$?

→ how decay α_t?

→ how choose actions? - always choose a_0 (won't learn)

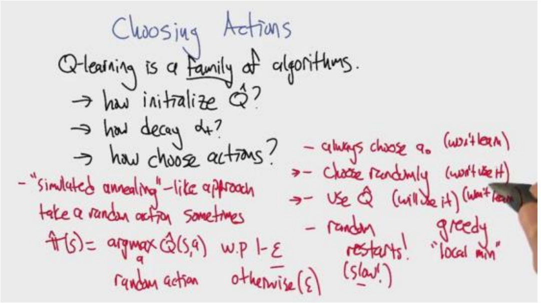
- choose randomly (won't use it)

- use $\hat{Q}$ (will use it) (won't learn)

$\forall s$ $\hat{Q}(s,a_0)$ = awesome (Chilean dollars) greedy

$\forall s, a \neq a_0$ $\hat{Q}(s,a)$ = terrible "local min"

- Q learning is a family of learning algorithms. It differs by 3 dimensions



Choosing Actions

Q-learning is a family of algorithms.

→ how initialize $\hat{Q}$?

→ how decay α_t?

→ how choose actions? - always choose a_0 (won't learn)

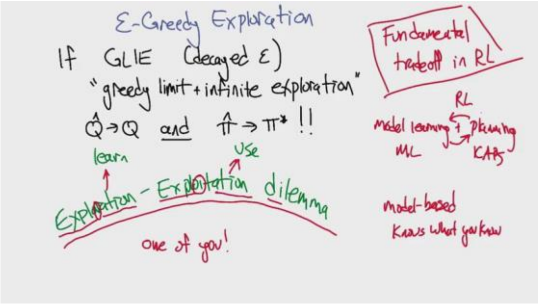
→- choose randomly (won't use it)

- "simulated annealing"-like approach →- use $\hat{Q}$ (will use it) (won't learn)

take a random action sometimes - random restarts! (slow!) greedy "local min"

$\hat{\pi}(s) = \arg\max_a \hat{Q}(s,a)$ w.p $1-\varepsilon$

random action otherwise (ε)



ε-Greedy Exploration

If GLIE (decayed ε)

"greedy limit + infinite exploration"

$\hat{Q} \rightarrow Q$ and $\hat{\pi} \rightarrow \pi^*$!!

learn use

Exploration - Exploitation dilemma

one of you!

Fundamental tradeoff in RL

RL

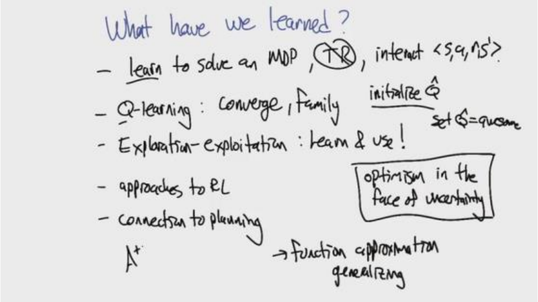
model learning ⇆ planning

ML KABS

model-based

knows what you know

- Simulated annealing: will random act with probability epsilon: will explore to all states



What have we learned?

- learn to solve an MDP, (T R), interact $\langle s,a,r,s' \rangle$

- Q-learning: converge, family initialize $\hat{Q}$

set $\hat{Q}$ = awesome

- Exploration-exploitation: learn & use!

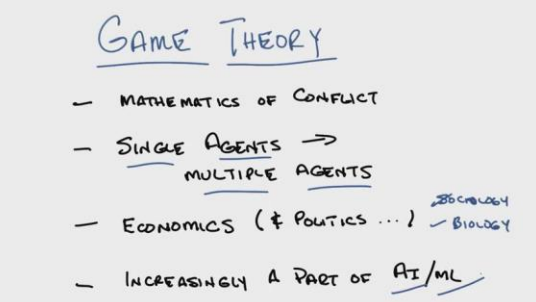
- approaches to RL optimism in the face of uncertainty

- connection to planning

A^* → function approximation

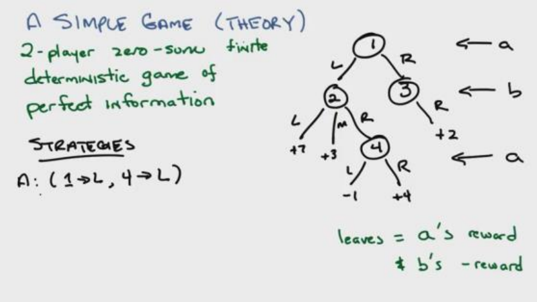
generalizing

RL 3 - Game Theory



GAME THEORY

- MATHEMATICS OF CONFLICT
- SINGLE AGENTS → MULTIPLE AGENTS
- ECONOMICS (& POLITICS ...) – SOCIOLOGY – BIOLOGY
- INCREASINGLY A PART OF AI/ML



A SIMPLE GAME (THEORY)

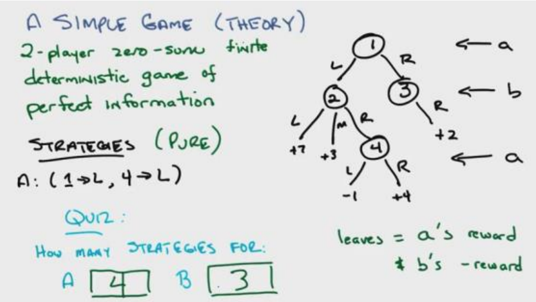
2-player zero-sum finite deterministic game of perfect information

STRATEGIES

A: (1→L, 4→L)

leaves = a's reward & b's -reward

- Game theory can be thought about in different perspectives, like listed
- Number is what assigned to a
- a got two actions: L and R, while b has 3 actions: L, M, R
- zero-sum: the sum of rewards is constant (not necessarily zero)
- no statistical transitions
- **strategies** (in game theory): mapping from state to actions (similar ideas as policy in MDP setting), always refer to one party (like a)



A SIMPLE GAME (THEORY)

2-player zero-sum finite deterministic game of perfect information

STRATEGIES (PURE)

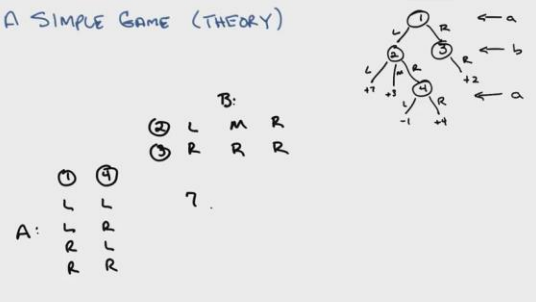
A: (1→L, 4→L)

QUIZ:

How many STRATEGIES FOR: A [4] B [3]

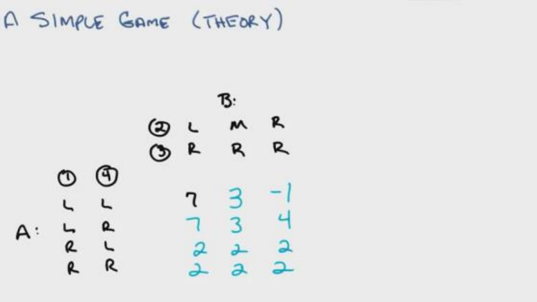
leaves = a's reward & b's -reward

- 4 choices for a, even though if I go R at stage 1, no need to make choice at state 4. But independently, there are 4 choices
- Strategies: what you might do in all states you might end up with



A SIMPLE GAME (THEORY)

			B:		
		②	L	M	R
		③	R	R	R
	①	④			
A:	L	L	7		
	L	R			
	R	L			
	R	R			

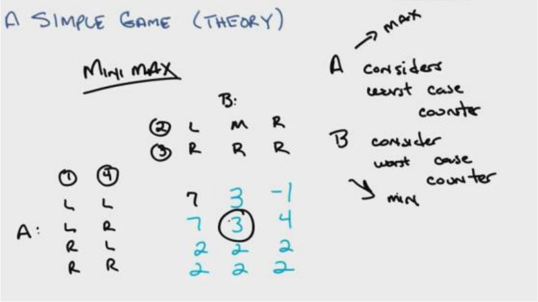


A SIMPLE GAME (THEORY)

			B:		
		②	L	M	R
		③	R	R	R
	①	④			
A:	L	L	7	3	-1
	L	R	7	3	4
	R	L	2	2	2
	R	R	2	2	2

- Form a matrix, whose value for A is the combined actions of A and B
- Once we write down the matrix, then nothing else matters. Then the graph can be removed. Everything about the game is captured in the matrix
- This is called the matrix form of the game. The L, R, M combinations do not matter as well. What matters is by following strategies, you got values XXX for A, and how you get there, does not matter

- RL is to optimize long-term reward -> pick a policy to have a best long term reward possible
- Here a has 4 strategies, while b has 3 strategies. Based on this matrix, what will a, and b take?
- The same minimax answer if we choose row first or column first under 2 player, zero-sum, deterministic perfect information game
- If you have the matrix, you can construct the tree based on it (may have multiple tress corresponding to the same matrix)



A SIMPLE GAME (THEORY)

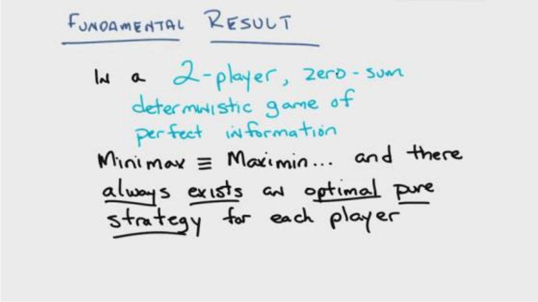
MINI MAX

B:

			②	L	M	R
			③	R	R	R
	①	④				
	L	L		7	3	-1
A:	L	R		7	3	4
	R	L		2	2	2
	R	R		2	2	2

A considers worst case counter ↗ max

B consider worst case counter ↘ min

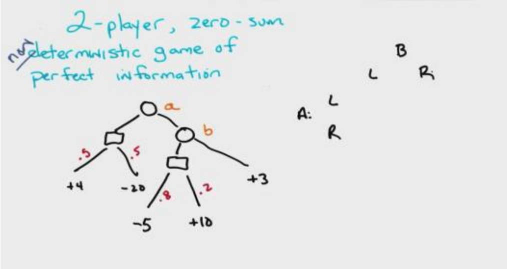


FUNDAMENTAL RESULT

In a 2-player, zero-sum deterministic game of perfect information

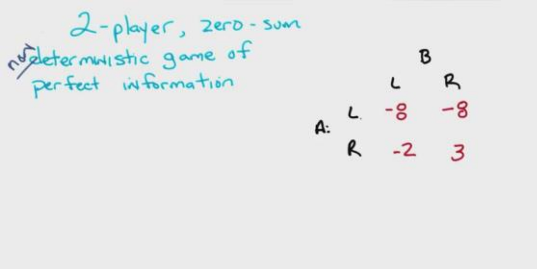
Minimax ≡ Maximin... and there always exists an optimal pure strategy for each player

- Minimax theory
- Assuming the other party take the counter strategy
- The value of this game for a is 4 if both players are rationale
- a first or b first, the results are the same
- Assumption: everyone is trying to maximize reward



2-player, zero-sum non deterministic game of perfect information

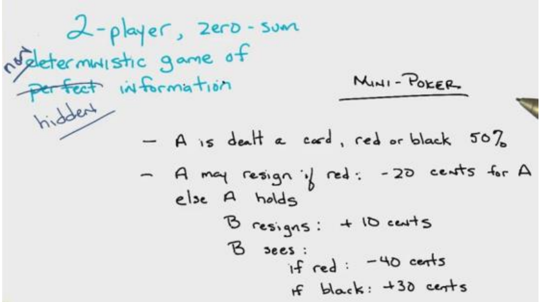
		B	
		L	R
A:	L		
	R		



2-player, zero-sum non deterministic game of perfect information

		B	
		L	R
A:	L	-8	-8
	R	-2	3

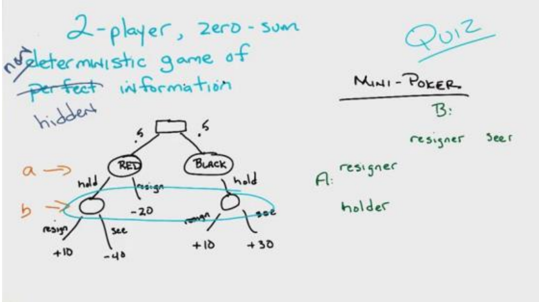
- Square box: stochastic rewards; Circle box: deterministic rewards
- The randomness does not necessarily happen to leaf only under general non-deterministic game
- We can reconstruct a tree based on the matrix, but infinitely many trees corresponds to the matrix
- The minimax theorem still holds tree for non-deterministic game (Von Neumann theorem)
- The only thing matters is the matrix, and use minimax or maxmini to figure out the solution



2-player, zero-sum non deterministic game of ~~perfect~~ hidden information

MINI-POKER

- A is dealt a card, red or black 50%
- A may resign if red: -20 cents for A
  else A holds
  - B resigns: + 10 cents
  - B sees:
    - if red: -40 cents
    - if black: +30 cents



2-player, zero-sum non deterministic game of ~~perfect~~ hidden information

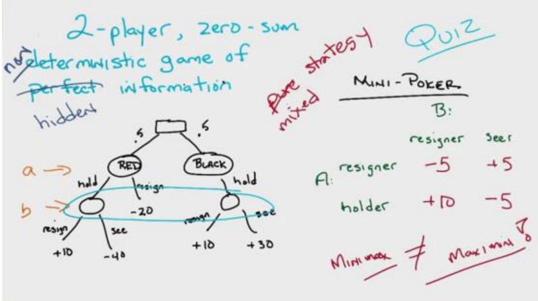
QUIZ

MINI-POKER

		B:	
		resigner	seer
A:	resigner		
	holder		

- Red is bad for A, while black is good for A

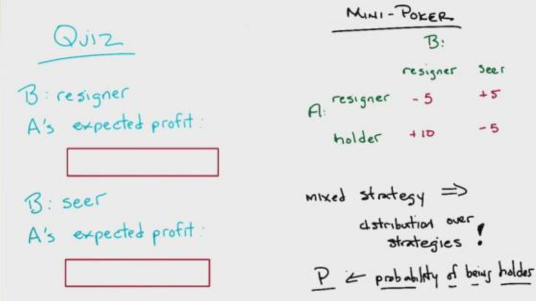
- Color of A is random, with 50% to be red and 50% to be black
- B doesn't know the result of A
- Hidden information, as B does not know the result of A
- A will not resign for black card
- B has not idea which state to be in (figure above right), hence hidden information
- A hold or resign based on red card (sure strategy for black card)
- B will resign or see when A choose to hold (2 strategies)



Mini-Poker

		B: resigner	B: seer
A:	resigner	−5	+5
	holder	+10	−5

- A resigner and B resigner, means when A got red, resign ->-20. But when A got black, always wins -> +10 => 0.5*(-20 + 10) = -5
- Hidden information, means the order of making choice make differences
- If A chooses first, -5; if B chooses first, +5
- If one is consistent over his actions, then the other party can manipulate it and earn profits <- why order of choosing matters
- Pure strategy (or consistent strategy) -> mixed strategy (distribution over strategies. I.e. 30% holder, and 70% resigner)


Quiz

B: resigner
A's expected profit:

B: seer
A's expected profit:

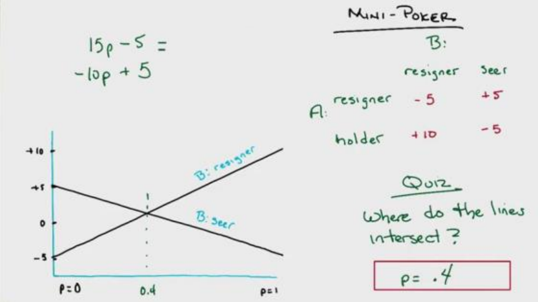
Mini-Poker

		B: resigner	B: seer
A:	resigner	-5	+5
	holder	+10	-5

mixed strategy ⇒ distribution over strategies!

P ← probability of being holder

- Quiz 1: B always resigner, and what's A's expected profit given probability of being A holder



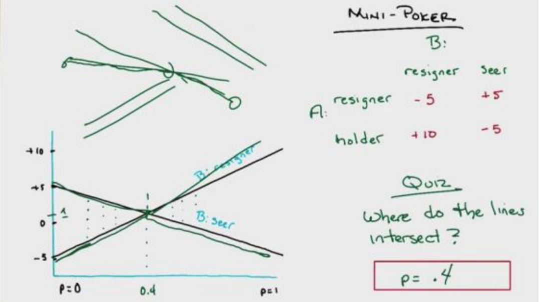
Mini-Poker

		B: resigner	B: seer
A:	resigner	-5	+5
	holder	+10	-5

Quiz

Where do the lines intersect?

p= .4



Mini-Poker

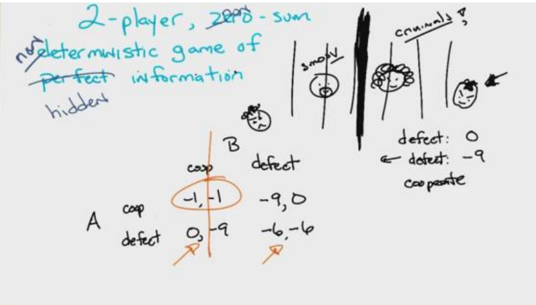
		B: resigner	B: seer
A:	resigner	-5	+5
	holder	+10	-5

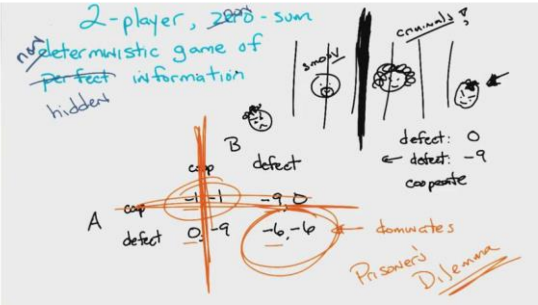
Quiz

Where do the lines intersect?

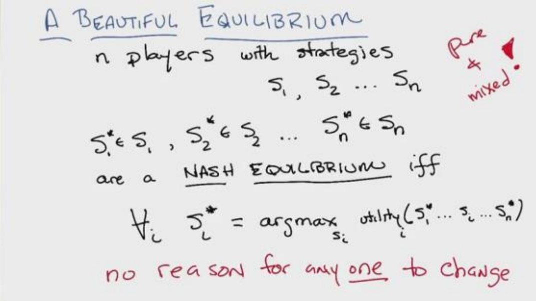
p= .4

- Two lines corresponding to the two strategies by B
- If A choose a mixed strategy with probability 0.4 choosing holder, and 0.6 choosing resigner, then the expected payoff is +1 to A, irrespective of what B chooses
- If B chooses to use mixed strategy as well, the final payoff will be the same as +1 to A
- If B chooses to use mixed strategy, payoff would be somewhere between these two lines
- The highlighted triangle (above right) is the payoff of A choosing different probabilities, and we choose to take the maximum
- If the triangle is decreasing (see the cross above), we will choose the left end point, instead of the cross point as the solution
- If B choose to use mixed strategy, and then consider A, the results would be the same as if we let A choose mixed strategy, and then consider B





- For non-zero sum game, need to put down the payoff for both players, instead of one number
- Based on the matrix, the best solution is both corporate and sit in jail for 1 month each
- Not joint choice, but individual choose
- But actually ended with both defect
- If A know B is going to corporate, then A should choose defect to save 1 month;
- If B know A is going to defect, B should defect to save 3 months
- Eventually end up with both defect
- For each column, A is always better to defect than corporate (0 vs -1 and -6 vs -9), so A will never corporate



A BEAUTIFUL EQUILIBRIUM

n players with strategies $S_1, S_2 \dots S_n$

pure & mixed!

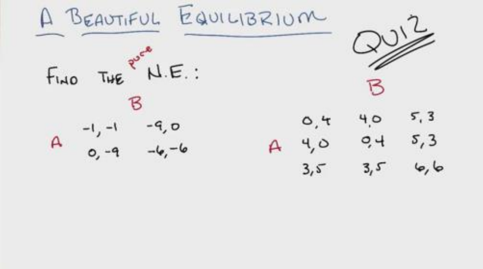
$s_1^* \in S_1, s_2^* \in S_2 \dots s_n^* \in S_n$

are a NASH EQUILIBRIUM iff

$$\forall_i \; s_i^* = \operatorname{argmax}_{s_i} \text{utility}_i(s_1^* \dots s_i \dots s_n^*)$$

no reason for any one to change

- In (pure) Nash equilibrium, no player would like to change their strategies for better payoff (assuming all the other keep constant)



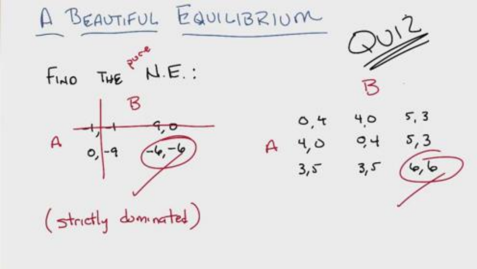
A BEAUTIFUL EQUILIBRIUM

QUIZ

FIND THE pure N.E.:

		B	
A	-1,-1	-9,0	
	0,-9	-6,-6	

		B	
	0,4	4,0	5,3
A	4,0	0,4	5,3
	3,5	3,5	6,6



A BEAUTIFUL EQUILIBRIUM

QUIZ

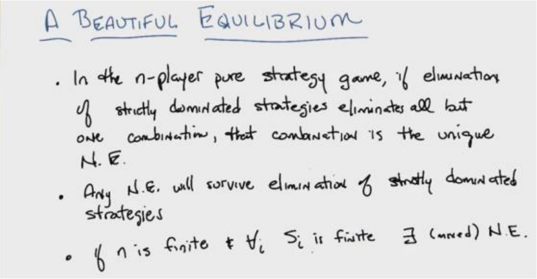
FIND THE pure N.E.:

		B
A	~~-1,-1~~	~~-9,0~~
	~~0,-9~~	-6,-6

(strictly dominated)

		B	
	0,4	4,0	5,3
A	4,0	0,4	5,3
	3,5	3,5	6,6

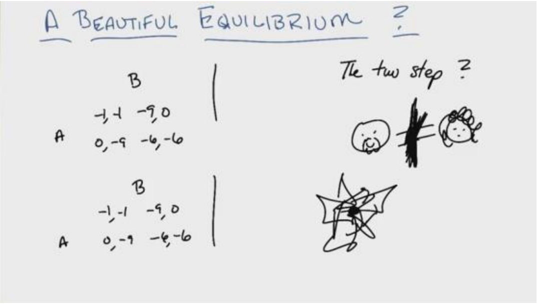
- In the first case, the second row strictly dominate the first row
- In the second case, both have the largest payoff



A BEAUTIFUL EQUILIBRIUM

- In the n-player pure strategy game, if elimination of strictly dominated strategies eliminates all but one combination, that combination is the unique N. E.
- Any N.E. will survive elimination of strictly dominated strategies
- if n is finite & $\forall_i$ S_i is finite $\exists$ (mixed) N.E.

- Listed are 3 fundamental theorems for Nash Equilibrium
- 1st: in iterative fashion: it may not dominate in one iteration. Could do row&col iterations
- 3rd: for finite game, there exist at least 1 Nash Equilibrium (could be mixed)

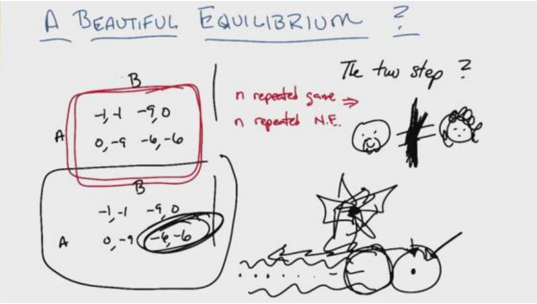


A BEAUTIFUL EQUILIBRIUM ?

The two step ?

		B
	-1,-1	-9,0
A	0,-9	-6,-6

		B
	-1,-1	-9,0
A	0,-9	-6,-6



A BEAUTIFUL EQUILIBRIUM ?

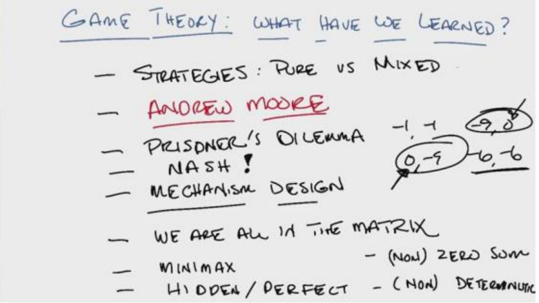
The two step ?

n repeated game ⇒ n repeated N.E.

		B
	-1,-1	-9,0
A	0,-9	-6,-6

		B
	-1,-1	-9,0
A	0,-9	-6,-6

- Get communication instead of wall may not solve the prisoner dilemma, because the person goes first, the person lose
- This happen only when the game plays once. If game happens more times => keep corporating together for ever; corporating will earn reward
- Suppose we play the game 2 times. For A, either we can set up like 4 strategies: either corporate or defect at each round; or we can let A's choice depends on B's choice in the first round, ending with 8 strategies
- For example, we play the game for 20 times. If we go forward, it might make sense to corporate. But going backward, at 20th game, whatever happen is the sunk cost. Final one is determined (-6, -6). At 19th game, 20th game result is there, so 19th game will both defect as well => even if we can play for many times, we will still end up at the same solution
- **Theorem**: for N repeated games, the solution is N repeated Nash Equilibriums. If a game has multiple N.E. point, the results could oscillating => once you are in one N.E. point, we are not going to leave it



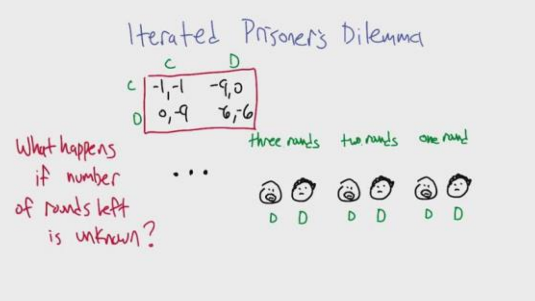
GAME THEORY: WHAT HAVE WE LEARNED?

- STRATEGIES: PURE VS MIXED
- ANDREW MOORE
- PRISONER'S DILEMMA
- NASH!
- MECHANISM DESIGN
- WE ARE ALL IN THE MATRIX
- MINIMAX
- HIDDEN / PERFECT

- (NON) ZERO SUM
- (NON) DETERMINISTIC

- **Mechanism Design**: to solve the prisoner dilemma, change the game! (change utility) 1) If A cares 50% of B staying the jail, then the payments shift; 2) the people defect, got punished. 3) hire policemen or hitters

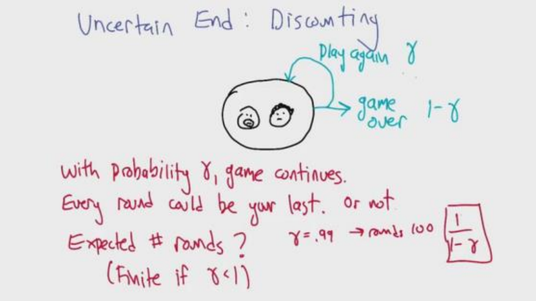
RL 4 - Game Theory Continued



Iterated Prisoner's Dilemma

$$\begin{array}{c|cc} & C & D \\ \hline C & -1,-1 & -9,0 \\ D & 0,-9 & -6,-6 \end{array}$$

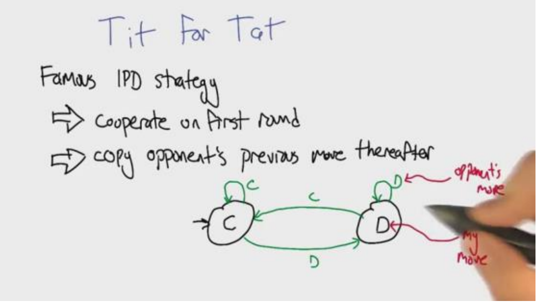
What happens if number of rounds left is unknown?



Uncertain End: Discounting

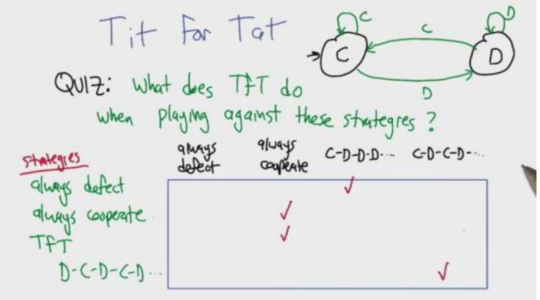
With probability γ, game continues.
Every round could be your last. Or not.
Expected # rounds? γ=.99 → rounds 100 $\boxed{\frac{1}{1-\gamma}}$
(Finite if γ<1)

- Conclusion: if the number of rounds is unknown, the results would be different
- Gamma here is the same as the discounting factor gamma



Tit For Tat

Famous IPD strategy
⇨ Cooperate on first round
⇨ copy opponent's previous move thereafter

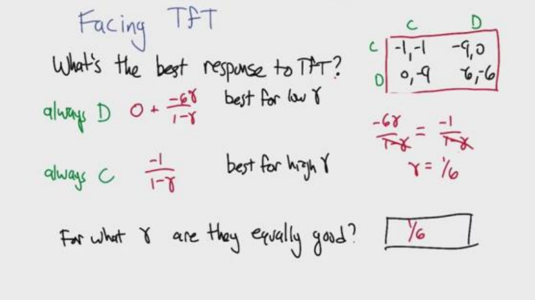


Tit For Tat

QUIZ: What does TFT do when playing against these strategies?

strategies	always defect	always cooperate	C-D-D-D···	C-D-C-D-···
always defect			✓	
always cooperate		✓		
TFT		✓		
D-C-D-C-D···				✓

- In uncertain number of rounds, we cannot write down strategies like C-D-C-D-D, or in the form of trees. We need some representations for unbound round of games
- Tit for Tat strategy



Facing TFT

$$\begin{array}{c|cc} & C & D \\ \hline C & -1,-1 & -9,0 \\ D & 0,-9 & -6,-6 \end{array}$$

What's the best response to TFT?

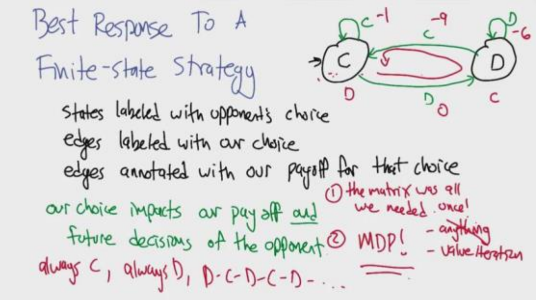
always D $0 + \frac{-6\gamma}{1-\gamma}$ best for low γ

always C $\frac{-1}{1-\gamma}$ best for high γ

$\frac{-6\gamma}{\cancel{1-\gamma}} = \frac{-1}{\cancel{1-\gamma}}$

γ = 1/6

For what γ are they equally good? $\boxed{1/6}$

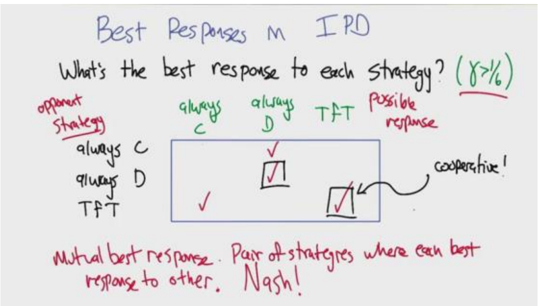


Best Response To A Finite-state Strategy

States labeled with opponent's choice
edges labeled with our choice
edges annotated with our payoff for that choice
our choice impacts our pay off and future decisions of the opponent
① the matrix was all we needed once!
② MDP!
- anything
- value iteration
always C, always D, D-C-D-C-D-...

- Listed are the expected payoff when we are dealing with Tit for Tat: always defect or always corporate

- For gamma < 1/6, we should always defect; while gamma > 1/6, we should corporate
- The matrix form is good if we only play once
- Use MDP to solve the strategy: mapping from state to action => only 3 results (policies): always corporate (good for high discount factor), always defect (good for low discount factor), or D-C-D-C…
- If this is MDP, then there is no history. When you are in C, there are only two choices: C or D; the remaining will require you remember what you have done steps ago
- MDP always has a Markov deterministic policy



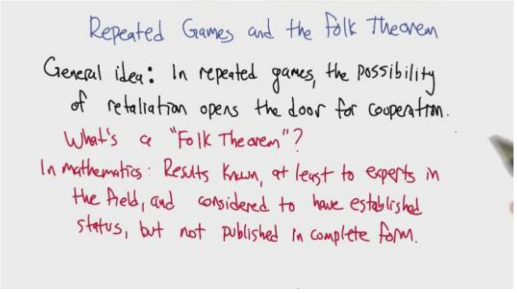
Best Responses in IPD

What's the best response to each strategy? ($\gamma > 1/6$)

opponent strategy	always C	always D	TFT	possible response
always C		✓		
always D		☑		
TFT	✓		☑	

Mutual best response. Pair of strategies where each best response to other. Nash!

- Note: we are not playing with Tit for Tat
- There are two Nash Equilibrium points: one of them is corporative

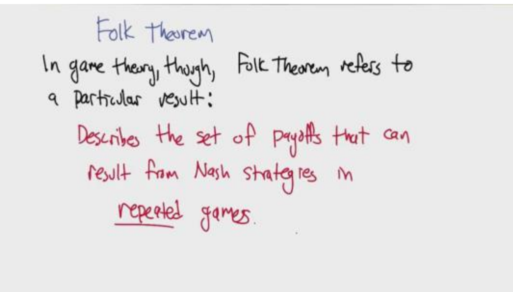


Repeated Games and the Folk Theorem

General idea: In repeated games, the possibility of retaliation opens the door for cooperation.

What's a "Folk Theorem"?

In mathematics: Results known, at least to experts in the field, and considered to have established status, but not published in complete form.

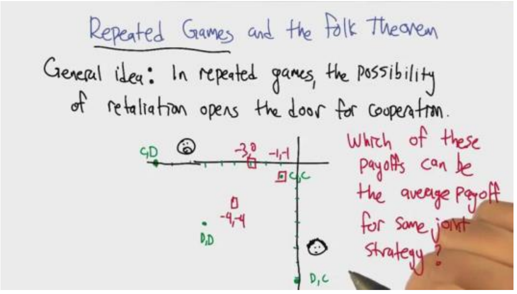


Folk theorem

In game theory, though, Folk Theorem refers to a particular result:

Describes the set of payoffs that can result from Nash strategies in repeated games.

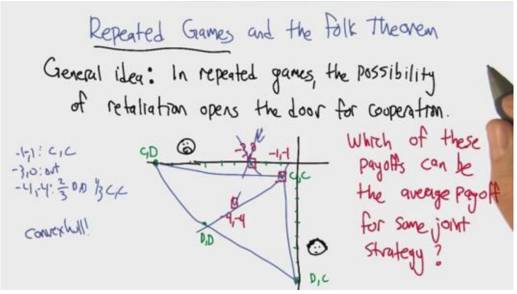
- Folk theorem: everybody can prove it, but not sure who proved it first



Repeated Games and the Folk Theorem

General idea: In repeated games, the possibility of retaliation opens the door for cooperation.

Which of these payoffs can be the average payoff for some joint strategy?

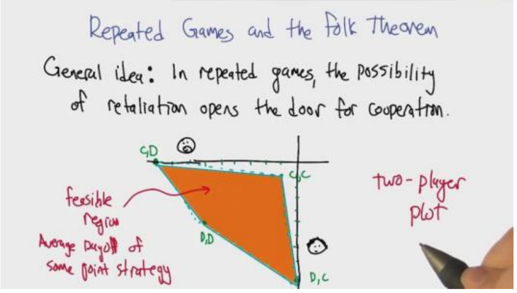


Repeated Games and the Folk Theorem

General idea: In repeated games, the possibility of retaliation opens the door for cooperation.

-1,-1: C,C
-3,0: out
-4,-4: $\frac{2}{3}$ D,D $\frac{1}{3}$ C,C

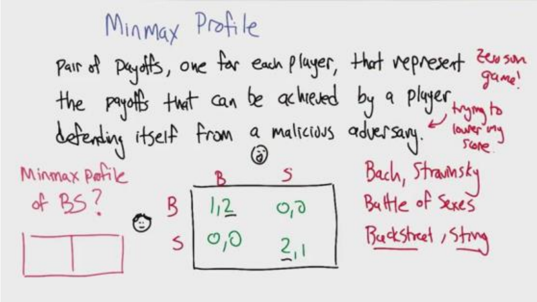
Which of these payoffs can be the average payoff for some joint strategy?



Repeated Games and the Folk Theorem

General idea: In repeated games, the possibility of retaliation opens the door for cooperation.

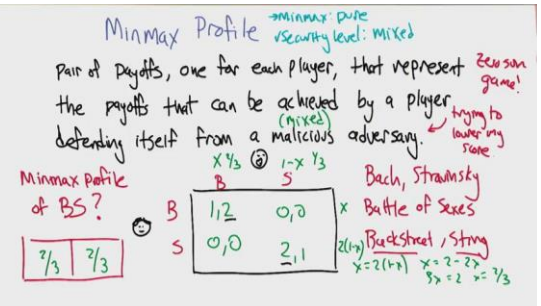
- For quiz: just to execute some joint strategies, and see the payoff. Not necessarily to be in Nash equilibrium



Minmax Profile

Pair of Payoffs, one for each player, that represent the payoffs that can be achieved by a player defending itself from a malicious adversary.

	B	S
B	1,2	0,0
S	0,0	2,1

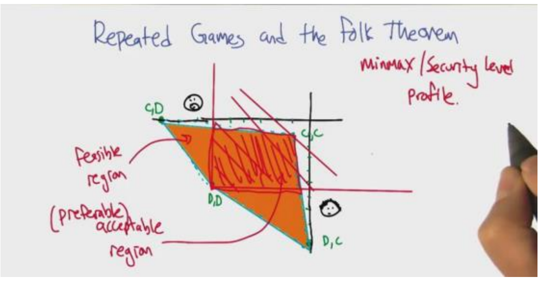


Minmax Profile

Pair of Payoffs, one for each player, that represent the payoffs that can be achieved by a player defending itself from a (mixed) malicious adversary.

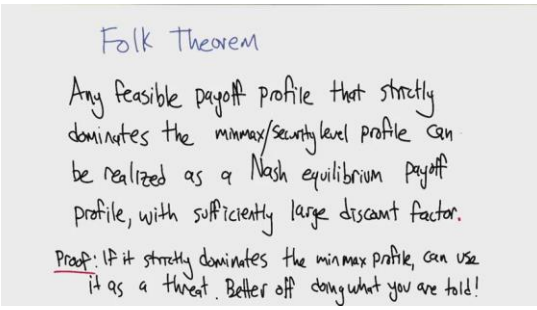
	B	S
B	1,2	0,0
S	0,0	2,1

- In the game, B and S tends to get the other lower score
- Smooth may choose mixed decision
- Minimax strategy = 1, while mixed strategy = 2/3



Repeated Games and the Folk Theorem

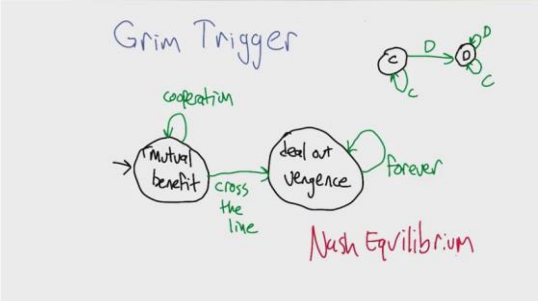
- Feasible region: possible payoff by using mixed strategies
- Acceptable region: both smooth and curly can get better than worst results



Folk Theorem

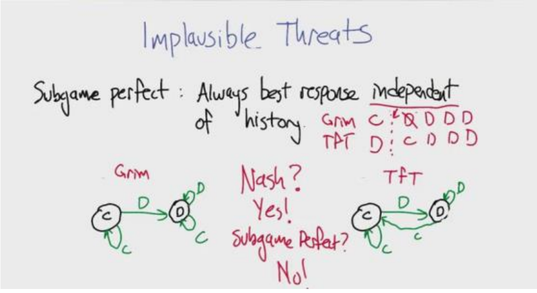
Any feasible payoff profile that strictly dominates the minmax/security level profile can be realized as a Nash equilibrium payoff profile, with sufficiently large discount factor.

Proof: If it strictly dominates the minmax profile, can use it as a threat. Better off doing what you are told!



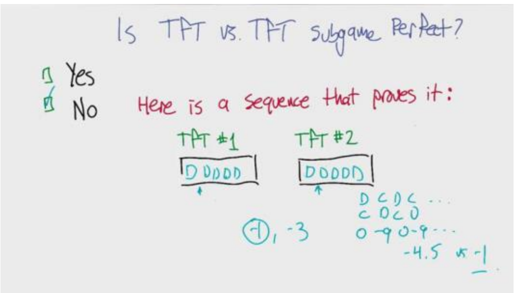
Grim Trigger

- Another perspective to prove Folk theorem is to use Grim Trigger
- Corporate (mutual benefit) until the cpty cross the line, and perform revenge for ever
- If A know B is taking the strategy above, then A will probably hold mutual benefit => no one has incentive to cross the line



Implausible Threats

Subgame perfect : Always best response independent of history.

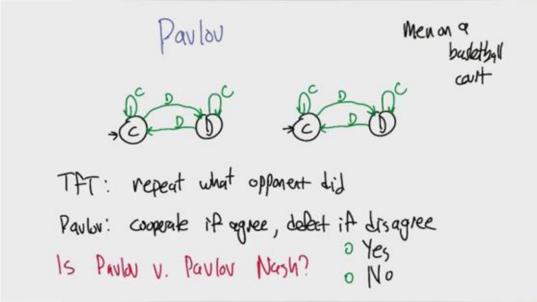


Is TFT vs. TFT Subgame Perfect?

☐ Yes
☑ No

Here is a sequence that proves it:

- Problem: in some case, the threats is implausible (too crazy, nobody will do it)
- Plausible treats will cause subgame perfect: each player always takes best response independent of history
- Could take a better action based on its strategy, given some history
- TFT vs TFT is not subgame perfect: suppose we start with (D,D), then it will go with (D,D) for ever. We can step in and change it to (C,C) to have a better payoff



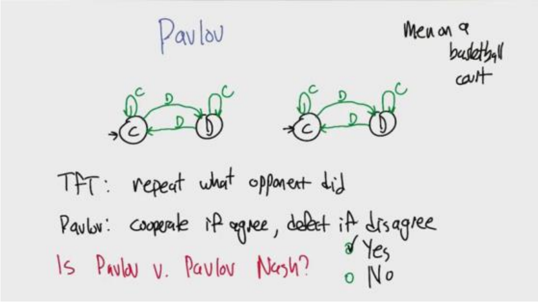
Pavlov

Men on a basketball court

TFT: repeat what opponent did

Pavlov: cooperate if agree, defect if disagree

Is Pavlov v. Pavlov Nash? ○ Yes ○ No



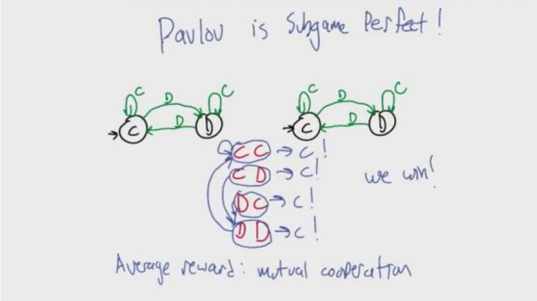
Pavlov

Men on a basketball court

TFT: repeat what opponent did

Pavlov: cooperate if agree, defect if disagree

Is Pavlov v. Pavlov Nash? ✓ Yes ○ No

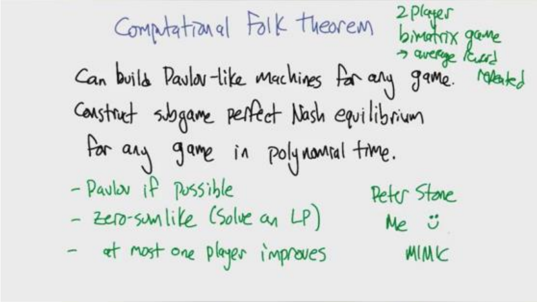


Pavlov is Subgame Perfect!

we win!

Average reward: mutual cooperation

- We can check four combination and see whether they are subgame perfect



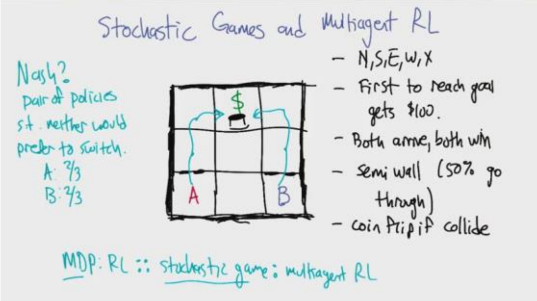
Computational Folk theorem

2 player bimatrix game → average reward repeated

Can build Pavlov-like machines for any game.

Construct subgame perfect Nash equilibrium for any game in polynomial time.

- Pavlov if possible — Peter Stone
- zero-sum like (Solve an LP) — Me ☺
- at most one player improves — MIMIC

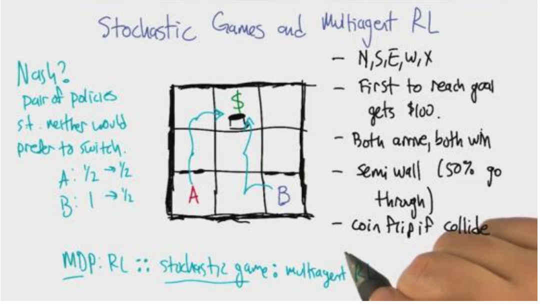


Stochastic Games and Multiagent RL

Nash? pair of policies s.t. neither would prefer to switch.
A: 2/3
B: 2/3

- N, S, E, W, X
- First to reach goal gets $100.
- Both arrive, both win
- semi wall (50% go through)
- coin flip if collide

MDP: RL :: stochastic game: multiagent RL



Stochastic Games and Multiagent RL

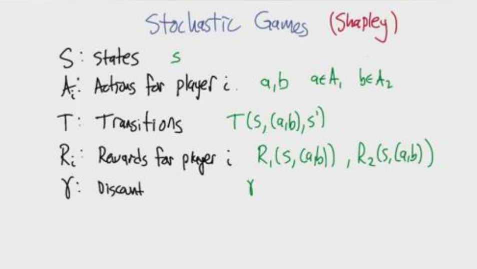
Nash? pair of policies s.t. neither would prefer to switch.
A: 1/2 → 1/2
B: 1 → 1/2

- N, S, E, W, X
- First to reach goal gets $100.
- Both arrive, both win
- semi wall (50% go through)
- coin flip if collide

MDP: RL :: stochastic game: multiagent RL

- MDP & RL is like stochastic game & multi-agent RL
- Dash line is the semi-wall, meaning if A would like to go through it, 50% will go through and 50% will stay put

- Goal: to get dollar
- They cannot occupy the same place; once one player reach the dollar sign, the game is over
- If they both go to the same position, they will throw a coin, and decide who will occupy the position
- When B change the strategy from going north to going west, then A’s chance stay the same, while B’s chance decreases. This is Nash equilibrium. The same for A going east, while B going north
- Both going to center is not Nash equilibrium, as you can go north



Stochastic Games (Shapley)

S: States s

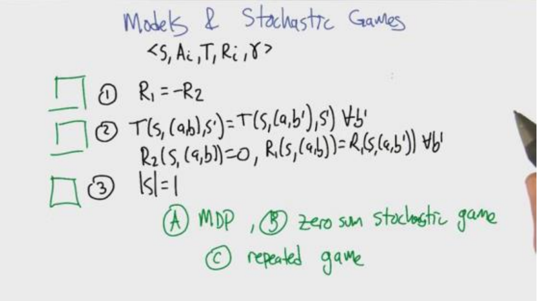
A_i: Actions for player i a, b $a \in A_1$ $b \in A_2$

T: Transitions $T(s,(a,b),s')$

R_i: Rewards for player i $R_1(s,(a,b))$, $R_2(s,(a,b))$

γ: Discount γ

- Stochastic game setup



Models & Stochastic Games

$\langle S, A_i, T, R_i, \gamma \rangle$

□ ① $R_1 = -R_2$

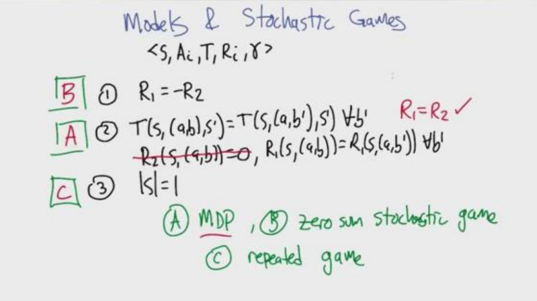
□ ② $T(s,(a,b),s') = T(s,(a,b'),s')\ \forall b'$

$R_2(s,(a,b)) = 0$, $R_1(s,(a,b)) = R_1(s,(a,b'))\ \forall b'$

□ ③ $|S| = 1$

Ⓐ MDP, Ⓑ zero sum stochastic game

Ⓒ repeated game



Models & Stochastic Games

$\langle S, A_i, T, R_i, \gamma \rangle$

B ① $R_1 = -R_2$

A ② $T(s,(a,b),s') = T(s,(a,b'),s')\ \forall b'$ $R_1 = R_2$ ✓

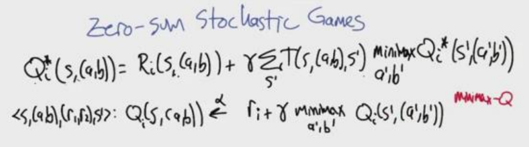
~~$R_2(s,(a,b)) = 0$~~, $R_1(s,(a,b)) = R_1(s,(a,b'))\ \forall b'$

C ③ $|S| = 1$

Ⓐ MDP, Ⓑ zero sum stochastic game

Ⓒ repeated game

- Markov if the results do not depend on the other player
- Only one state in 3

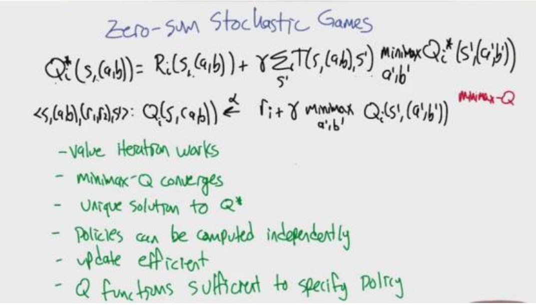


Zero-sum Stochastic Games

$$Q_i^*(s,(a,b)) = R_i(s,(a,b)) + \gamma \sum_{s'} T(s,(a,b),s') \underset{a',b'}{\text{minimax}}\, Q_i^*(s',(a',b'))$$

$$\langle s,(a,b),(r_1,r_2),s' \rangle : Q_i(s,(a,b)) \xleftarrow{\alpha} r_i + \gamma \underset{a',b'}{\text{minimax}}\, Q_i(s',(a',b'))$$ minimax-Q

- The maximum is taken under the assumption that everyone is trying to benefit me => works for team game, or somebody is willing to sacrifices themselves => therefore change max to minimax



Zero-sum Stochastic Games

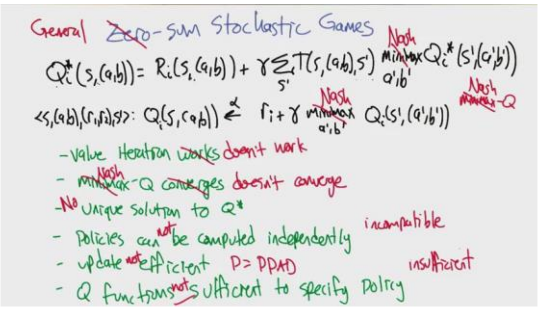
$$Q_i^*(s,(a,b)) = R_i(s,(a,b)) + \gamma \sum_{s'} T(s,(a,b),s') \underset{a',b'}{\text{minimax}}\, Q_i^*(s',(a',b'))$$

$$\langle s,(a,b),(r_1,r_2),s' \rangle : Q_i(s,(a,b)) \xleftarrow{\alpha} r_i + \gamma \underset{a',b'}{\text{minimax}}\, Q_i(s',(a',b'))$$ minimax-Q

- value iteration works
- minimax-Q converges
- unique solution to Q*
- policies can be computed independently
- update efficient
- Q functions sufficient to specify policy

- Minimax can be solved using linear programming
- Getting the second agent does not change much the problem
- One agenda: solved in polynomial time; while in stochastic game, it’s not known to solve using

polynomial time



General ~~Zero~~-sum Stochastic Games

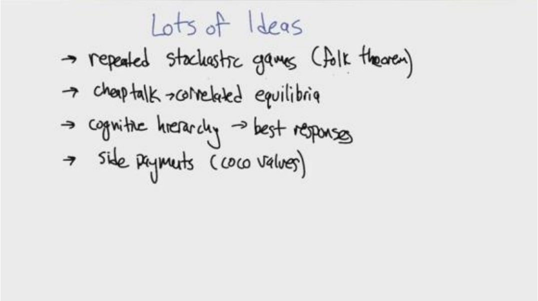
$Q_i^*(s,(a,b)) = R_i(s,(a,b)) + \gamma \sum_{s'} T(s,(a,b),s') \underset{a',b'}{\overset{\text{Nash}}{\cancel{\text{minimax}}}} Q_i^*(s',(a',b'))$

$\langle s,(a,b),(r_1,r_2),s'\rangle: Q_i(s,(a,b)) \xleftarrow{\alpha} r_i + \gamma \underset{a',b'}{\overset{\text{Nash}}{\cancel{\text{minimax}}}} Q_i(s',(a',b'))$

Nash ~~minimax~~-Q

- value iteration ~~works~~ doesn't work
- Nash ~~minimax~~-Q ~~converges~~ doesn't converge
- No unique solution to Q*
- Policies can not be computed independently   incompatible
- update not efficient   P=PPAD   insufficient
- Q functions not sufficient to specify policy

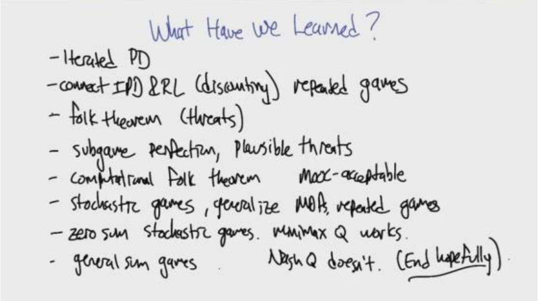
- Generate the framework to general stochastic game (not zero sum), then minimax operator will be replaced by Nash, as minimax does not suite the general case
- Value iteration does not work: if you loop through, weird thing will happen; does not necessarily converge and solve the equation
- Different Nash equilibrium may have different values => no unique solution
- Nash is defined using joint behavior, cannot be computed independently
- Two half Nash equilibrium cannot be Nash equilibrium => incompatible
- PPAD is known to be similar to NP, very hard to solve
- Based on Q function, cannot find what to behave => insufficient information



Lots of Ideas

→ repeated stochastic games (folk theorem)
→ cheap talk → correlated equilibria
→ cognitive hierarchy → best responses
→ side payments (coco values)

- Cheap talk: give the opportunity to let player to corporate a little bit
- Correlated equilibrium: version of Nash equilibria, more efficient to compute
- Give some of the reward back to you => to encourage corporate



What Have We Learned?

- Iterated PD
- connect IPD & RL (discounting) repeated games
- folk theorem (threats)
- subgame perfection, plausible threats
- computational folk theorem   max-acceptable
- stochastic games, generalize MDP, repeated games
- zero sum stochastic games. minimax Q works.
- general sum games   Nash Q doesn't. (End hopefully).

- PD: Prisoner Dilemma