Dynamic Programming

for Markov Decision Processes

Thomas Moerland

Leiden University

Last Week: Value function

Last Week: Value function

The average cumulative reward we get from a certain state/action for a given policy

Last Week: Value function

The average cumulative reward we get from a certain state/action for a given policy

- Each policy **π** has its own value function.

Last Week: Value function

The average cumulative reward we get from a certain state/action for a given policy

- Each policy **π** has its own value function.
- Defined for states **$v^{\pi}(s)$** and state-actions **$q^{\pi}(s,a)$**

Last Week: Value function

The average cumulative reward we get from a certain state/action for a given policy

- Each policy **π** has its own value function.
- Defined for states $\mathbf{v^{\pi}(s)}$ and state-actions $\mathbf{q^{\pi}(s,a)}$
- There is only one optimal value function $\mathbf{v^{*}(s)}$ / $\mathbf{q^{*}(s,a)}$

Last Week: Value function

The average cumulative reward we get from a certain state/action for a given policy

- Each policy **π** has its own value function.
- Defined for states **$v^{\pi}(s)$** and state-actions **$q^{\pi}(s,a)$**
- There is only one optimal value function **$v^*(s)$** / **$q^*(s,a)$**
- We can get **$\pi^*(a|s)$** from **$v^*(s)$** / **$q^*(s,a)$** by acting greedily with respect to it (selecting the action with the highest value)

Today

Today

Policy

$\pi(a|s)$

Value function

$v^{\pi}(s) \longleftarrow \longrightarrow q^{\pi}(s,a)$

Today

Policy

$\pi(a|s)$

Value function

$v^{\pi}(s) \longleftarrow \longrightarrow q^{\pi}(s,a)$

I

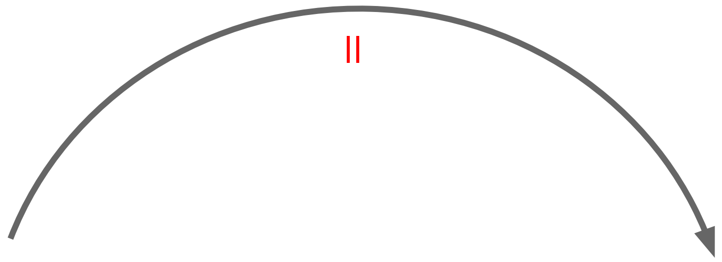
Discuss relations between values at different state(-actions)

[incl. recursion)

Today

Policy evaluation:

Compute the value of policy



Policy

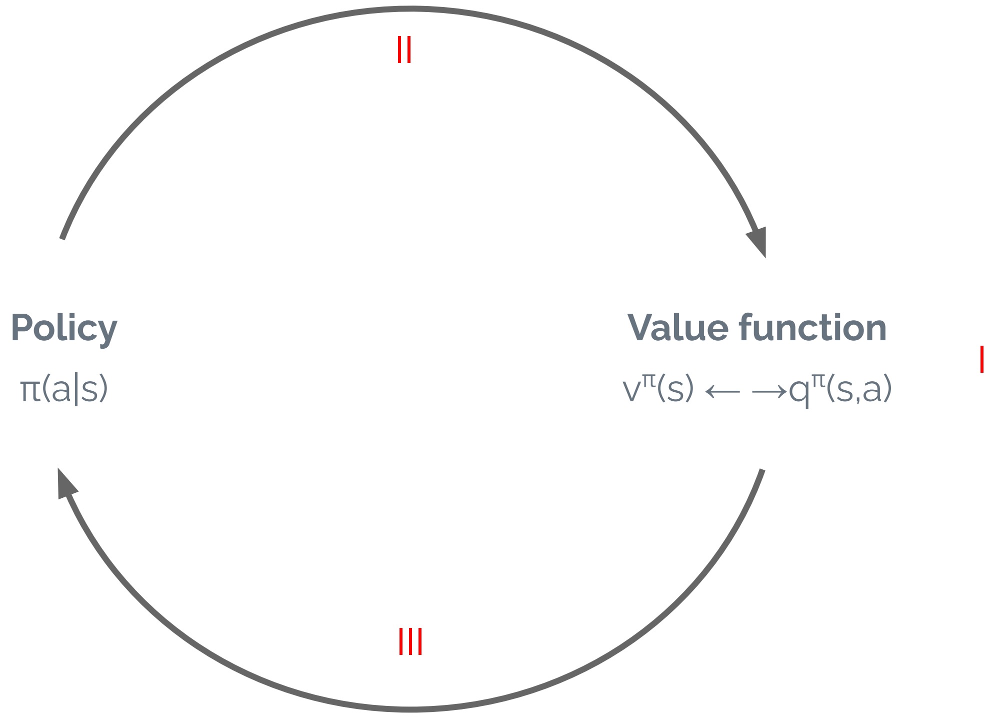
$\pi(a|s)$

Value function

I

$v^{\pi}(s) \longleftarrow \longrightarrow q^{\pi}(s,a)$

Today



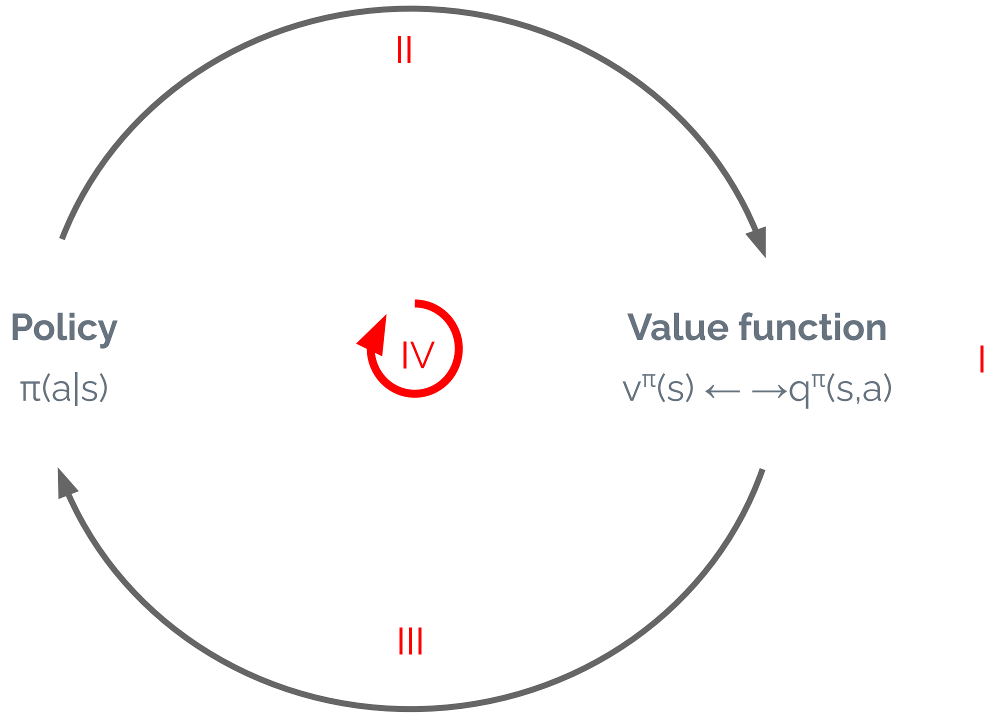
Implicit policies & Policy Improvement:

Define a new policy from a value function

Today

Generalized Policy Iteration:

Iterate both procedures to find the optimal value & policy



Overview

Overview

I. Value relations
   a. Relation between v(s) and q(s,a) — v(s) to q(s,a) & q(s,a) to v(s)
   b. Bellman Equation — v(s) to v(s') & q(s) to q(s')

Overview

I.	Value relations	
	a. Relation between v(s) and q(s,a)	v(s) to q(s,a) & q(s,a) to v(s)
	b. Bellman Equation	v(s) to v(s’) & q(s) to q(s’)
II.	Policy Evaluation (DP)	π to $v^{\pi}(s)$

Overview

I. Value relations	
a. Relation between v(s) and q(s,a)	v(s) to q(s,a) & q(s,a) to v(s)
b. Bellman Equation	v(s) to v(s') & q(s) to q(s')
II. Policy Evaluation (DP)	π to $v^{\pi}(s)$
III. Implicit policies	v(s)/q(s,a) to new π

Overview

I.	Value relations	
	a. Relation between v(s) and q(s,a)	v(s) to q(s,a) & q(s,a) to v(s)
	b. Bellman Equation	v(s) to v(s') & q(s) to q(s')
II.	Policy Evaluation (DP)	π to $v^{\pi}(s)$
III.	Implicit policies	v(s)/q(s,a) to new π
IV.	Finding the optimal value function & policy (v^*, q^*, π^*)	
	a. Bellman Optimality Equation	$v^*(s')$ to $v^*(s)$ & $q^*(s,a)$ to $q^*(s',a')$
	b. Value Iteration (DP)	
	c. Generalized Policy Iteration	
	d. Policy Iteration (DP)	

Part I

Value relations

Part Ia

Relation between v(s) and q(s,a)

Relation between v(s) to q(s,a)

Relation between v(s) to q(s,a)

The state value v(s) and state-action value q(s,a) represent

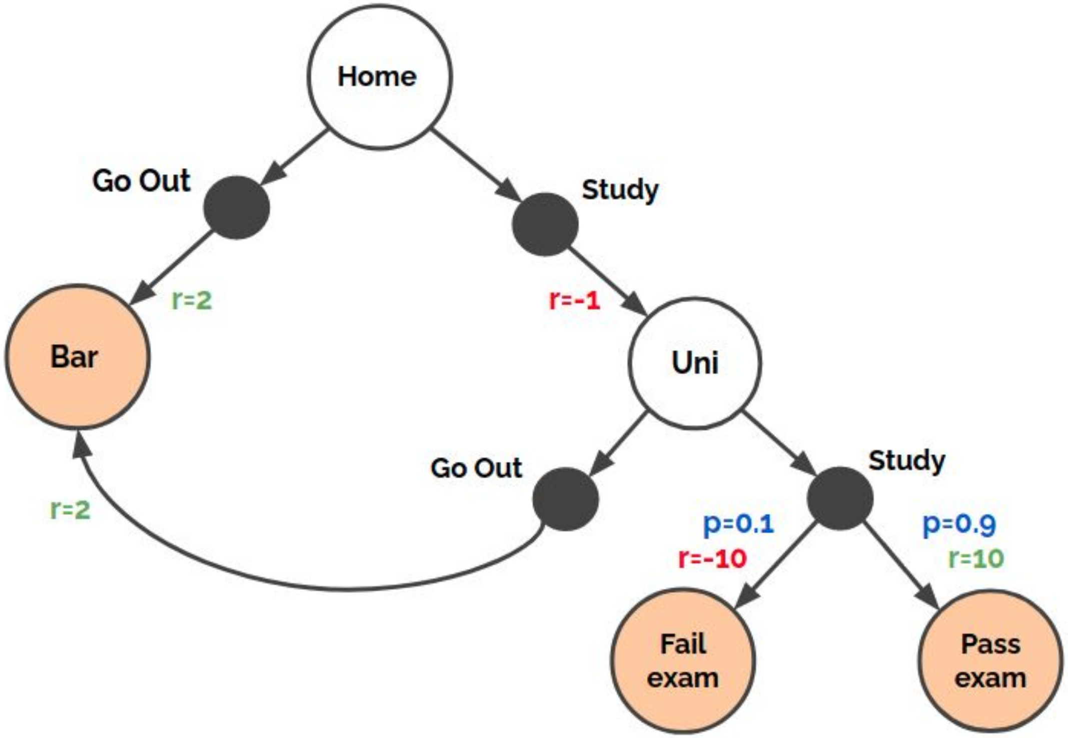
the *same* underlying function at different points

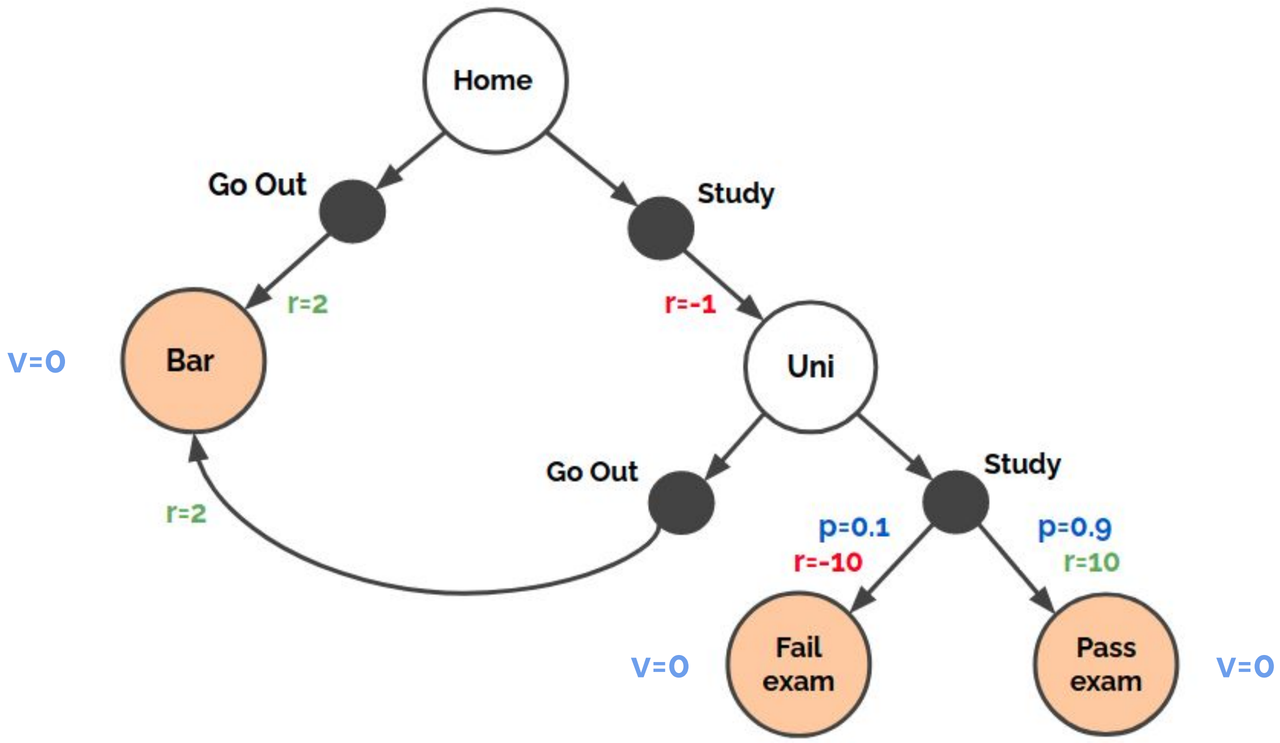
Relation between v(s) to q(s,a)

The state value v(s) and state-action value q(s,a) represent

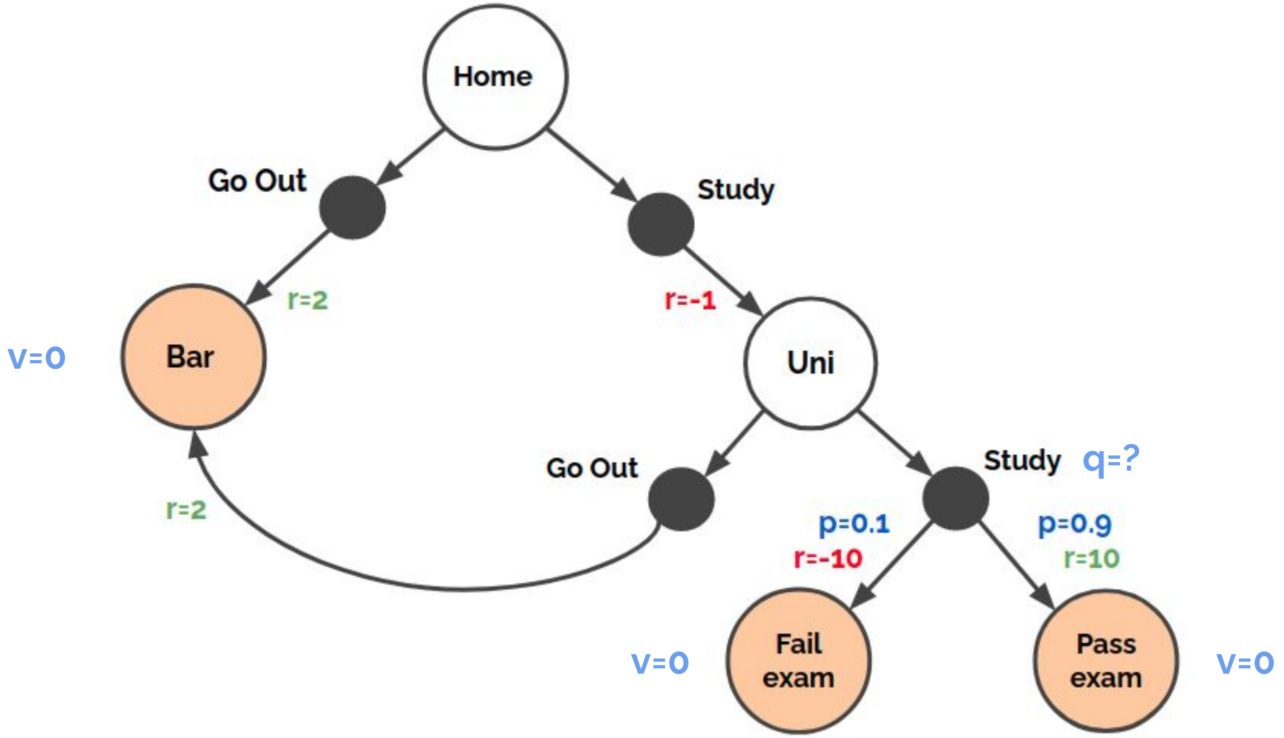
the *same* underlying function at different points

They can be rewritten into eachother!

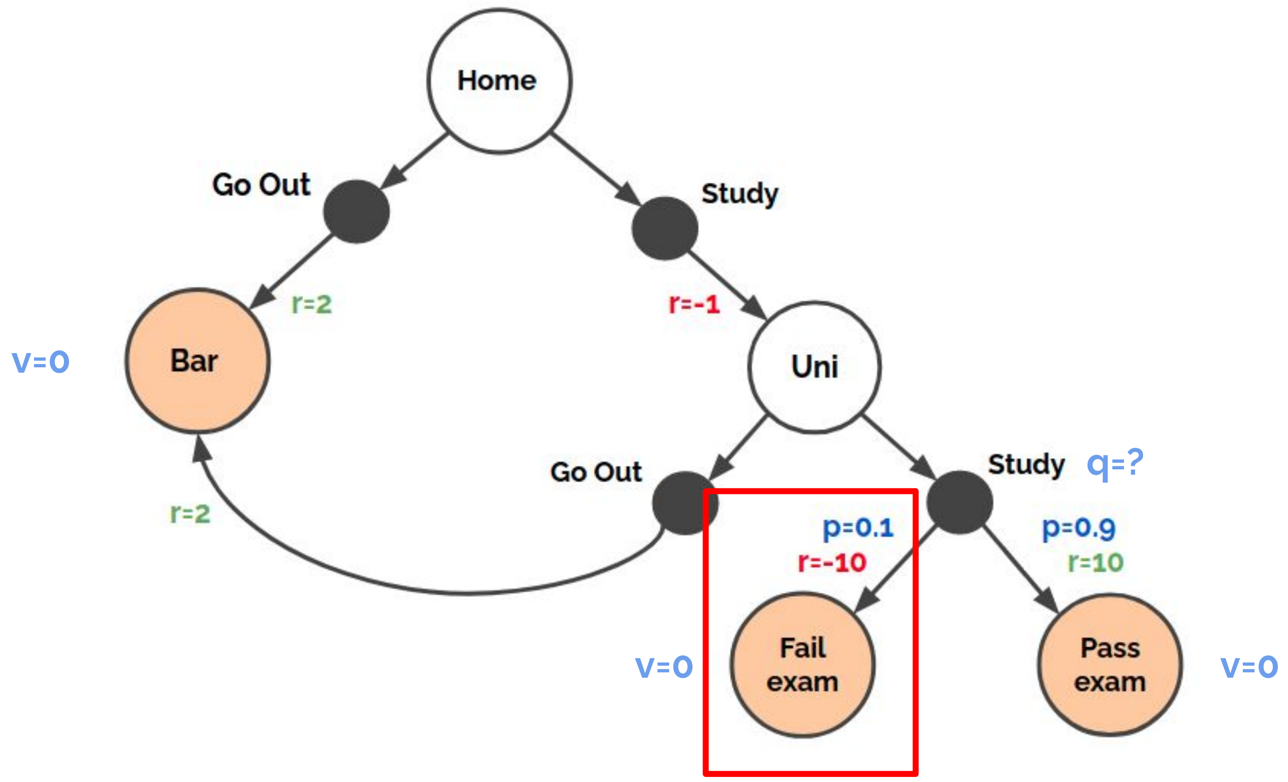

Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail exam
Pass exam



The value of terminal states is by definition 0.0

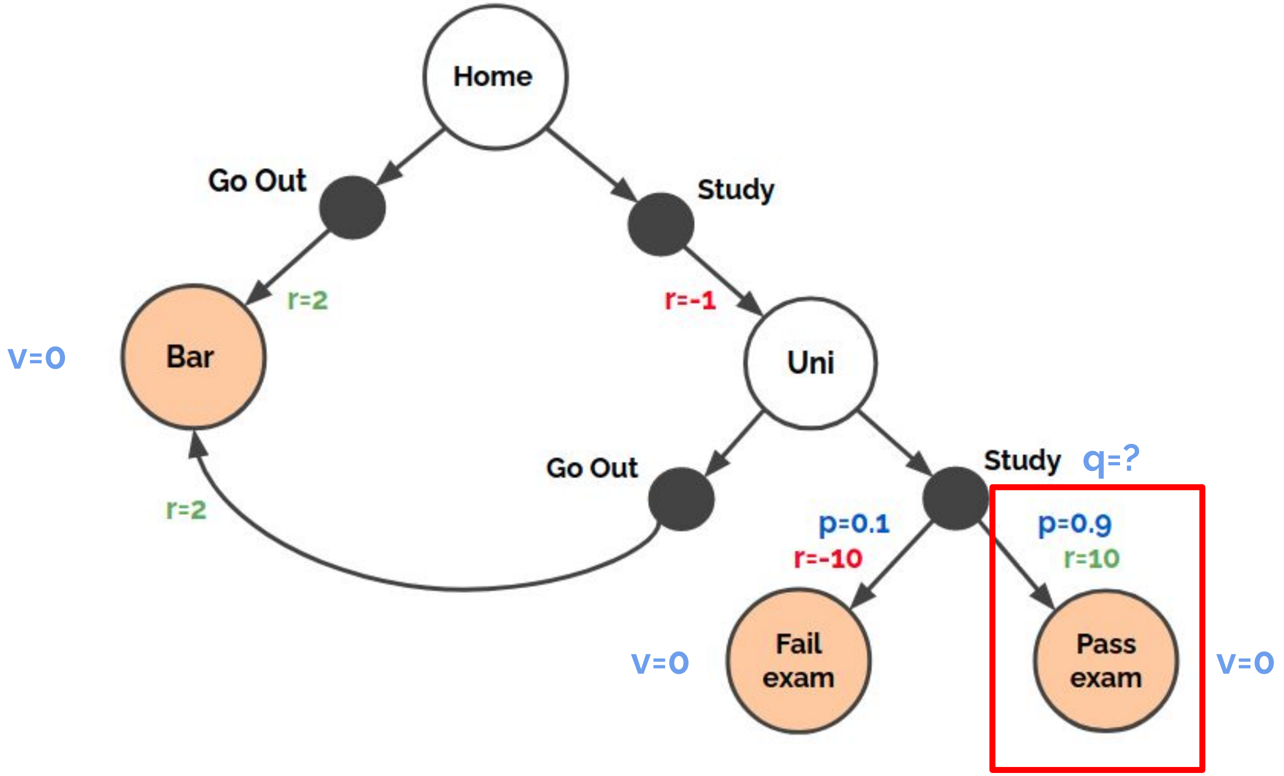


Question: What is q(Uni, Study)?



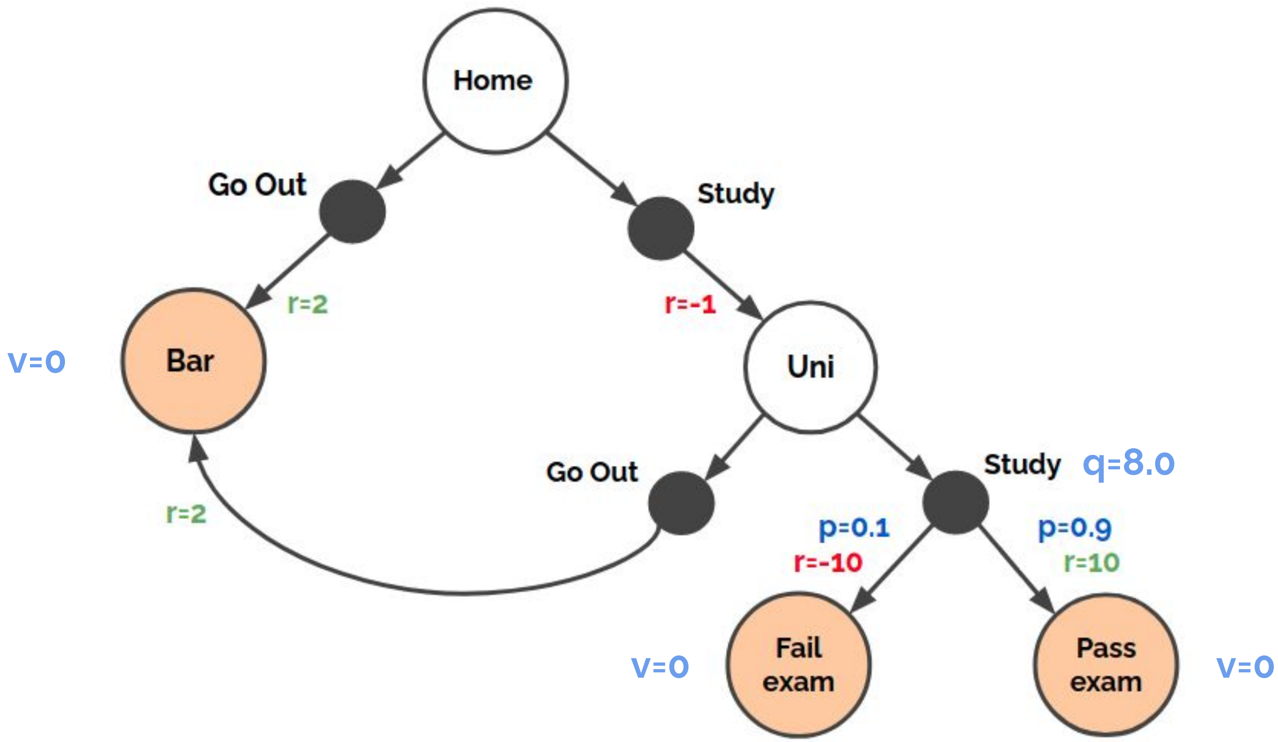
Question: What is q(Uni, Study)?

Answer: 10% of times we Fail Exam for r=-10 and v(s')=0 from next state,



Question: What is q(Uni, Study)?

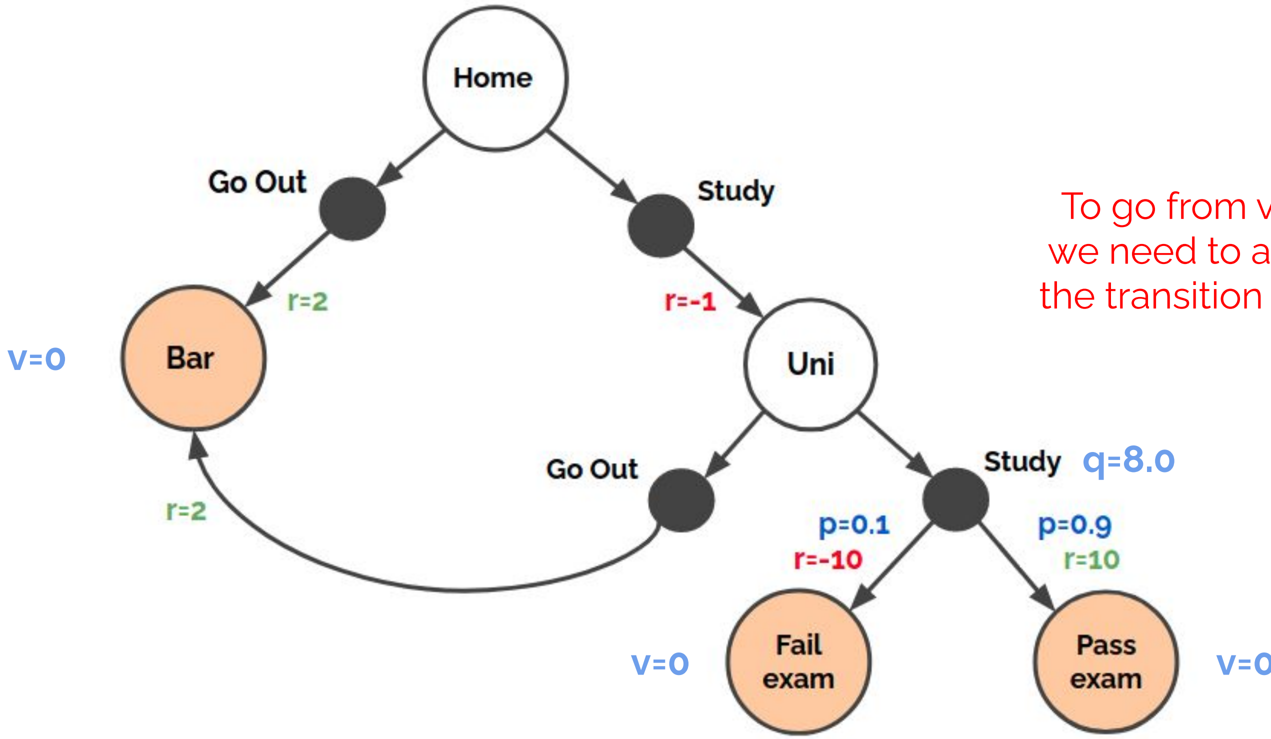
Answer: 10% of times we Fail Exam for r=-10 and v(s')=0 from next state,
90% of times we Pass Exam for r= 10 and v(s')=0 from next state



Question: What is q(Uni, Study)?

Answer: 10% of times we Fail Exam for r=-10 and v(s')=0 from next state,
90% of times we Pass Exam for r= 10 and v(s')=0 from next state

$q(Uni,Study) = 0.1 \cdot (10 + 0) + 0.9 \cdot (10 + 0) = 8.0$



To go from v(s') to q(s,a) we need to average over the transition probabilities

Question: What is q(Uni, Study)?

Answer: 10% of times we Fail Exam for r=-10 and v(s')=0 from next state,
90% of times we Pass Exam for r= 10 and v(s')=0 from next state

$q(\text{Uni},\text{Study}) = 0.1 \cdot (10 + 0) + 0.9 \cdot (10 + 0) = 8.0$

From v(s) to q(s,a)

From v(s) to q(s,a)

To get **q(s,a)** we weight the reward plus next state value **v(s')** by their transition probabilities

From v(s) to q(s,a)

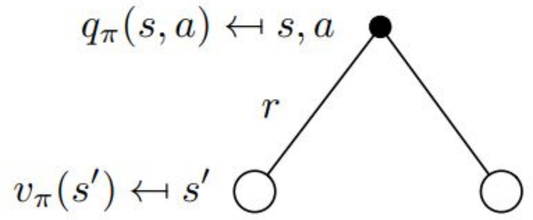
To get **q(s,a)** we weight the reward plus next state value **v(s')** by their transition probabilities

$$q^{\pi}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot v^{\pi}(s)\Big]$$

From v(s) to q(s,a)

To get **q(s,a)** we weight the reward plus next state value **v(s')** by their transition probabilities

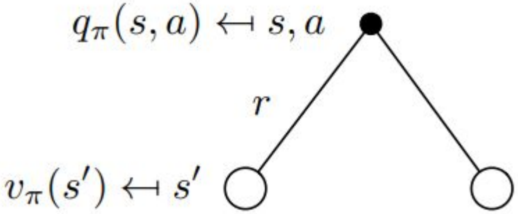
$$q^{\pi}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)} \Big[r(s,a,s') + \gamma \cdot v^{\pi}(s) \Big]$$



From v(s) to q(s,a)

To get **q(s,a)** we weight the reward plus next state value **v(s')** by their transition probabilities

$$q^{\pi}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot v^{\pi}(s)\Big]$$



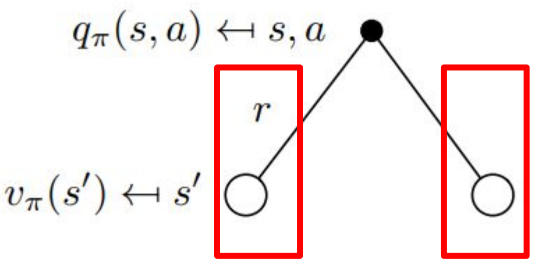
'Back-up diagram':

Visual illustration of a back-up formula

From v(s) to q(s,a)

To get **q(s,a)** we weight the reward plus next state value **v(s')** by their transition probabilities

$$q^\pi(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot v^\pi(s)\right]$$

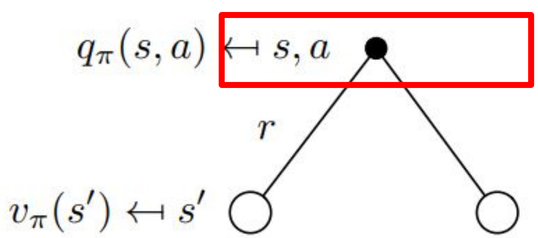


For each possible next state compute the reward plus next state value

From v(s) to q(s,a)

To get **q(s,a)** we weight the reward plus next state value **v(s')** by their transition probabilities

$$q^\pi(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot v^\pi(s) \right]$$

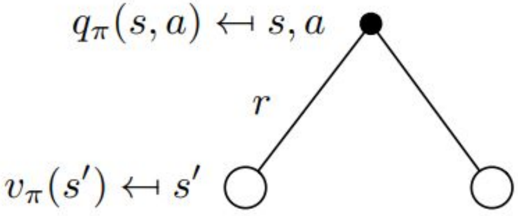


Average these according to their transition probabilities

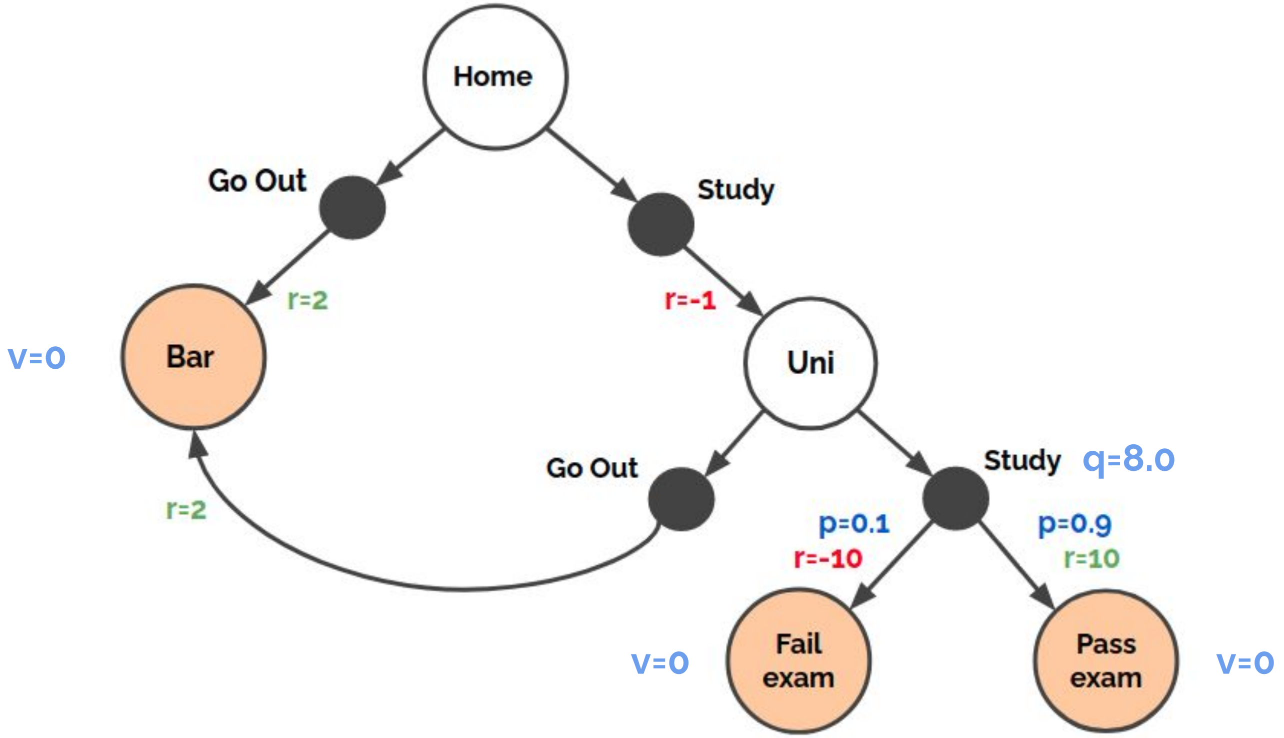
From v(s) to q(s,a)

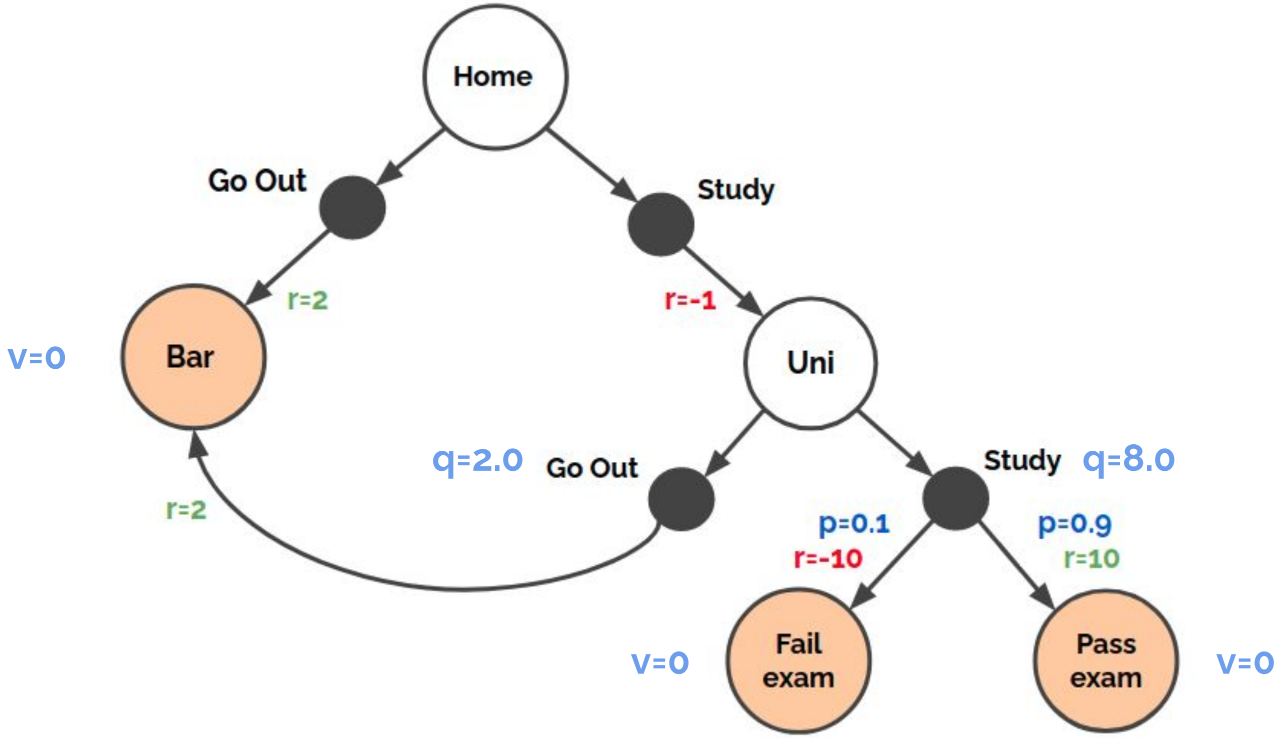
To get **q(s,a)** we weight the reward plus next state value **v(s')** by their transition probabilities

$$q^{\pi}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot v^{\pi}(s)\Big]$$

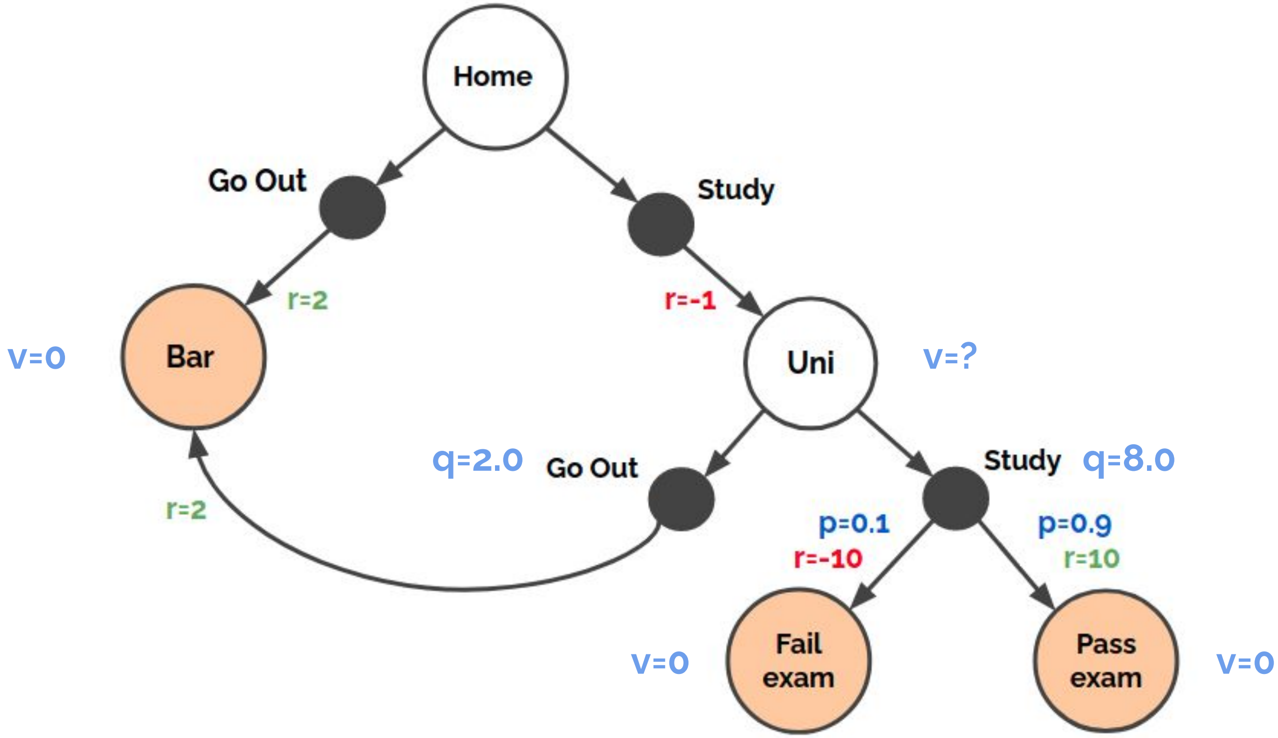


(requirement: transition probabilities, rewards, discount γ)

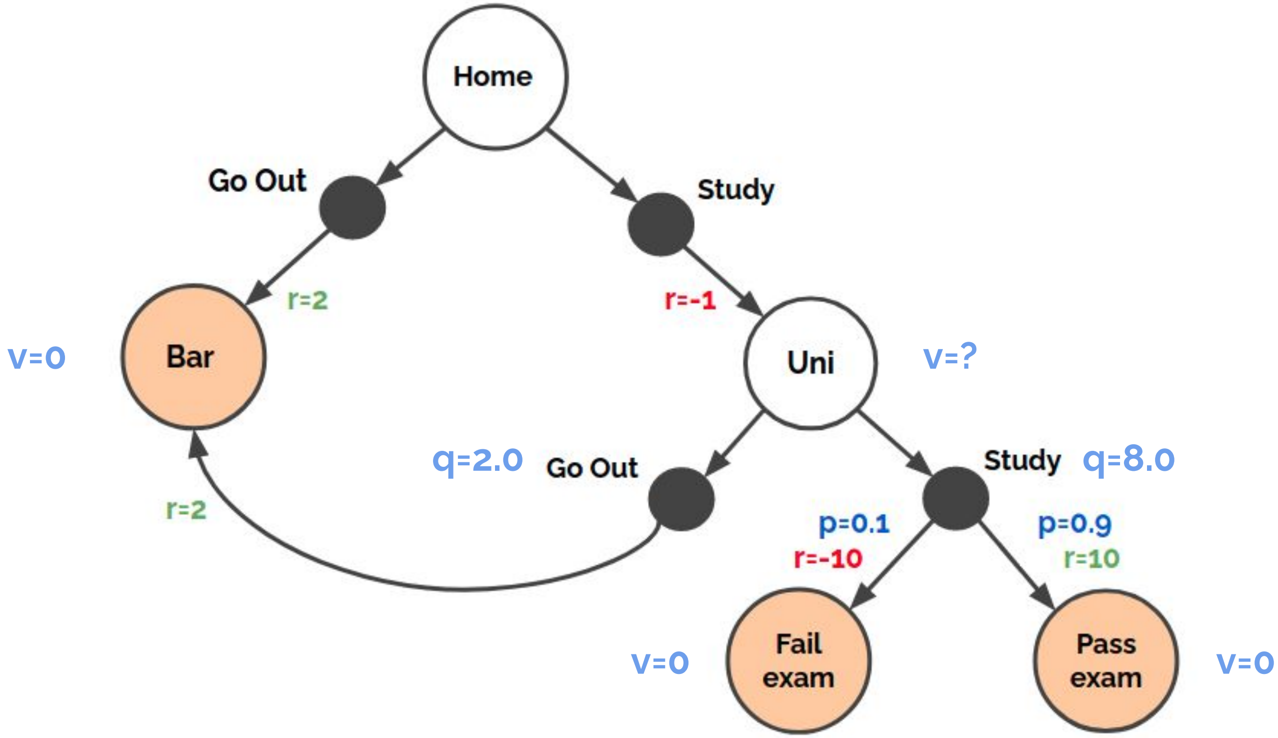

Home
Go Out
Study
r=2
r=-1
v=0
Bar
Uni
Go Out
Study
q=8.0
r=2
p=0.1
r=-10
p=0.9
r=10
v=0
Fail exam
Pass exam
v=0



q(Uni,Go Out) = 2.0 since we always get r=2.0 and terminate in the Bar

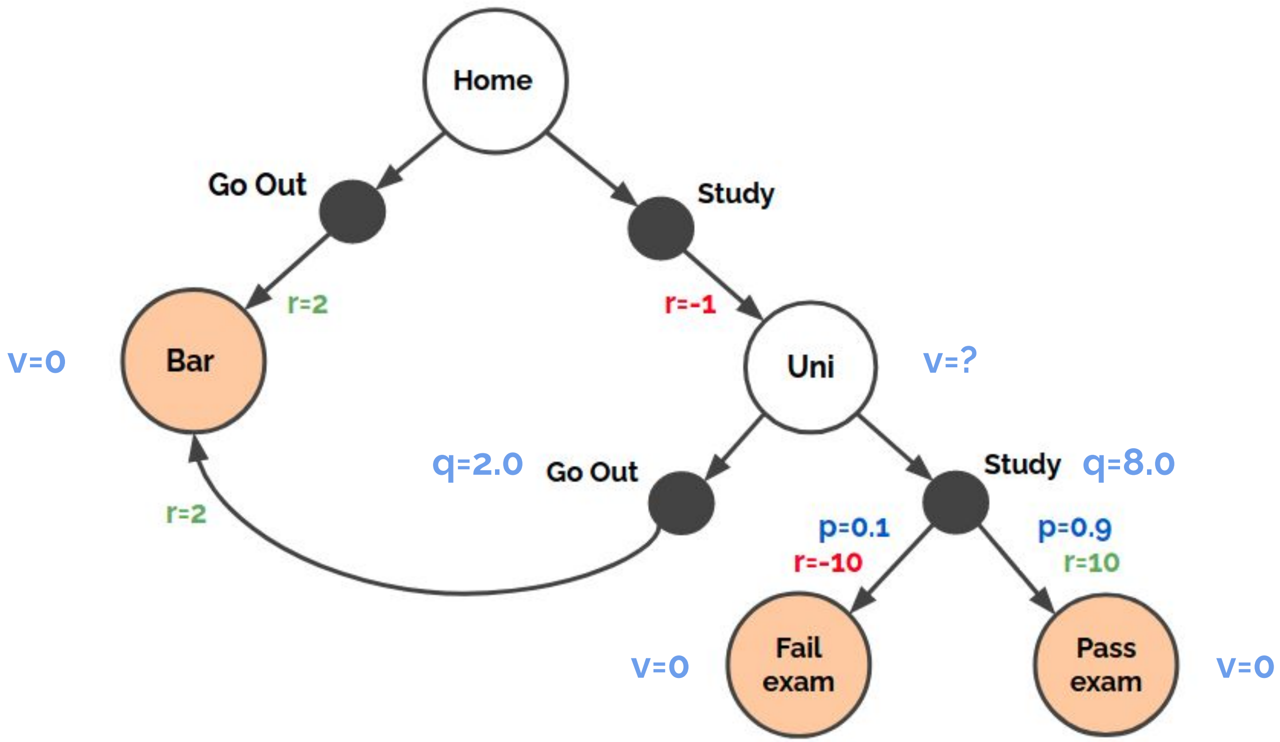


Question: But what is v(Uni)? (How do we go from q(s,a) to v(s)...)

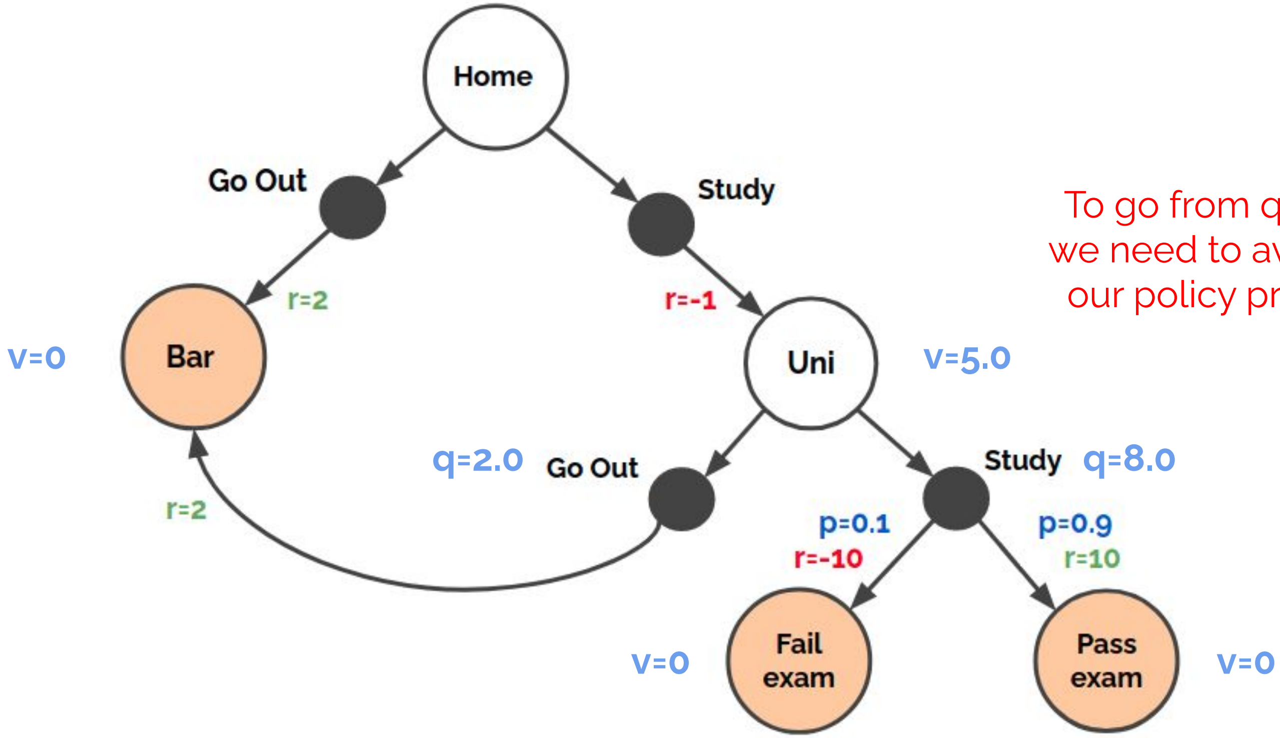


Question: But what is v(Uni)? (How do we go from q(s,a) to v(s)...)

Answer: Depends on our own policy!



Question: But what is v(Uni) under a *random* policy?



To go from q(s,a) to v(s) we need to average over our policy probabilities

Question: But what is v(Uni) under a *random* policy?

Answer: 50% of times we Go Out for an expected return of 2.0
50% of times we Study for an expected return of 8.0

$v(Uni) = 0.5 \cdot 2.0 + 0.5 \cdot 8.0 = 5.0$

From q(s,a) to v(s)

From q(s,a) to v(s)

To get **v(s)** we weight the **q(s,a)** of each available action by its selection probability

From q(s,a) to v(s)

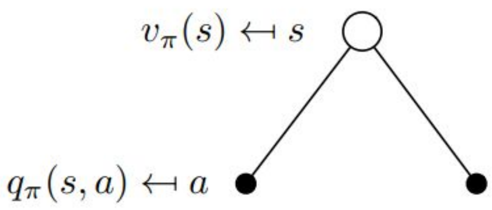
To get **v(s)** we weight the **q(s,a)** of each available action by its selection probability

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)}\Big[q^{\pi}(s, a)\Big]$$

From q(s,a) to v(s)

To get **v(s)** we weight the **q(s,a)** of each available action by its selection probability

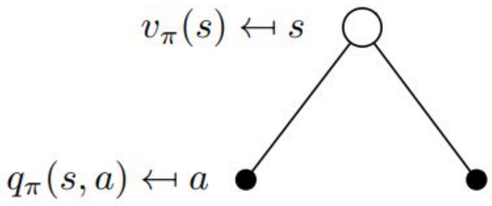
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \Big[q^{\pi}(s,a) \Big]$$



From q(s,a) to v(s)

To get **v(s)** we weight the **q(s,a)** of each available action by its selection probability

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)}\left[q^{\pi}(s, a)\right]$$

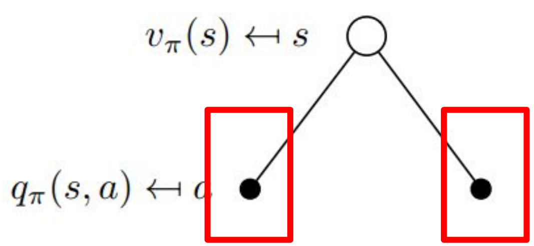


(*requirement*: policy probabilities)

From q(s,a) to v(s)

To get **v(s)** we weight the **q(s,a)** of each available action by its selection probability

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \boxed{q^{\pi}(s,a)}$$



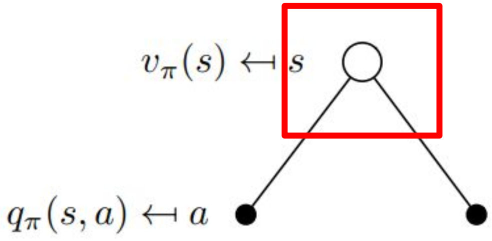
Take the state-action value of each available action…

(*requirement*: policy probabilities)

From q(s,a) to v(s)

To get **v(s)** we weight the **q(s,a)** of each available action by its selection probability

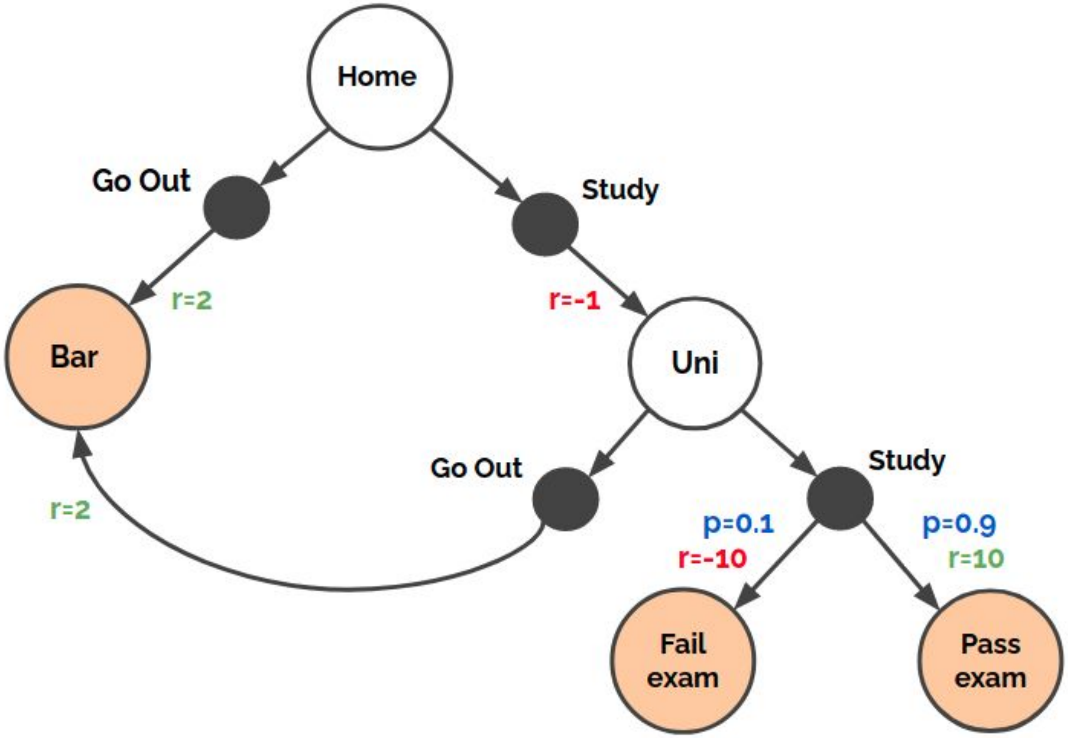
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)}\left[q^{\pi}(s,a)\right]$$



…and reweight them according to their policy probability

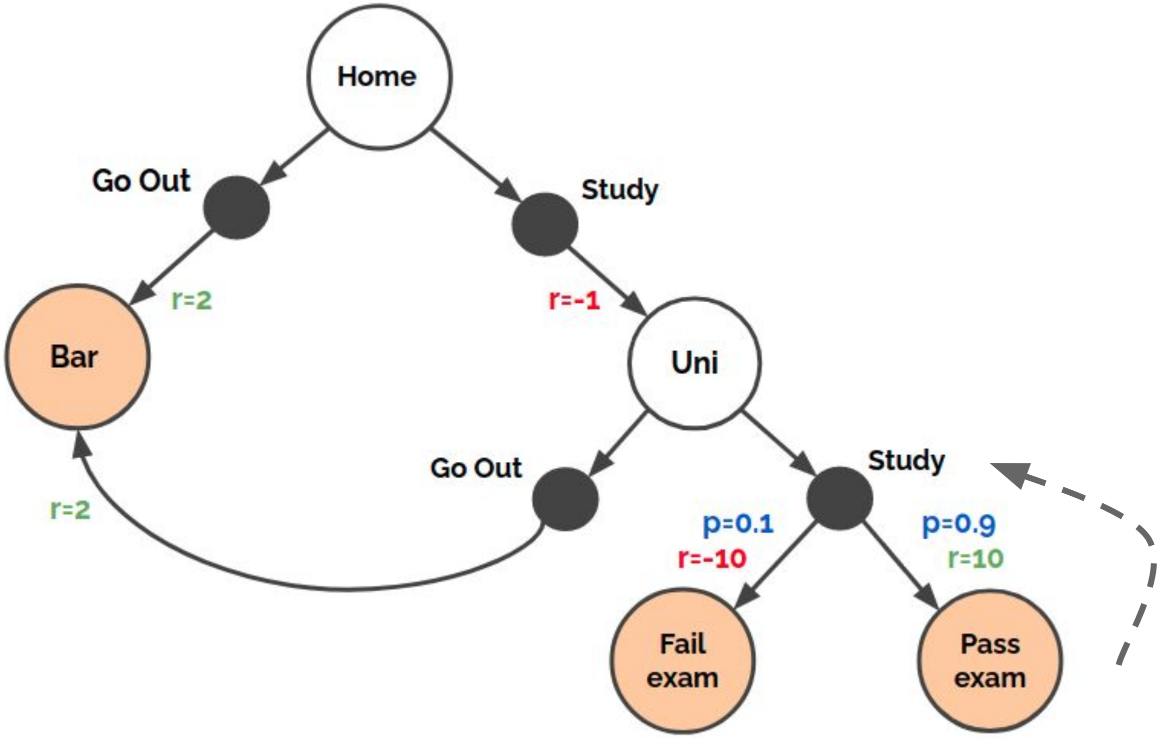
(*requirement*: policy probabilities)

Summary



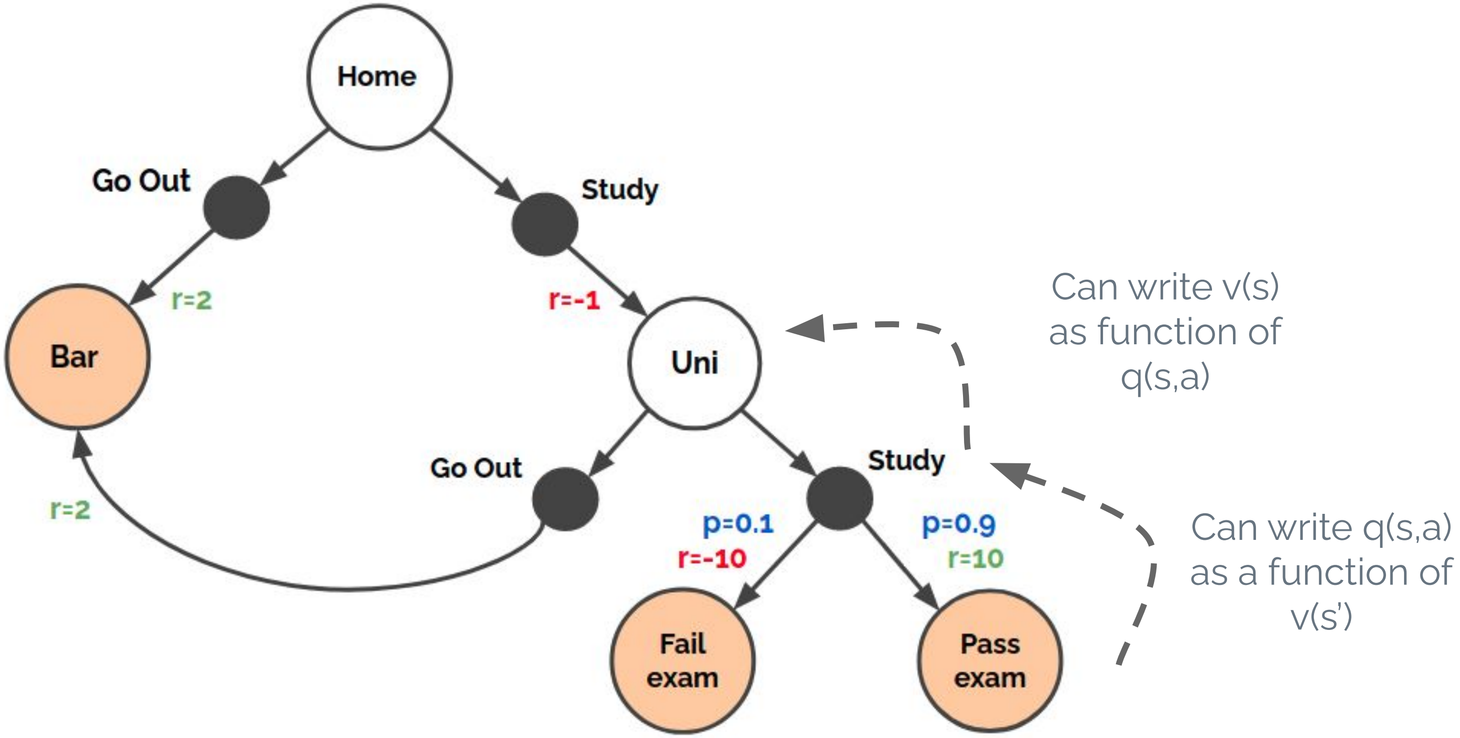
Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail exam
Pass exam

Summary



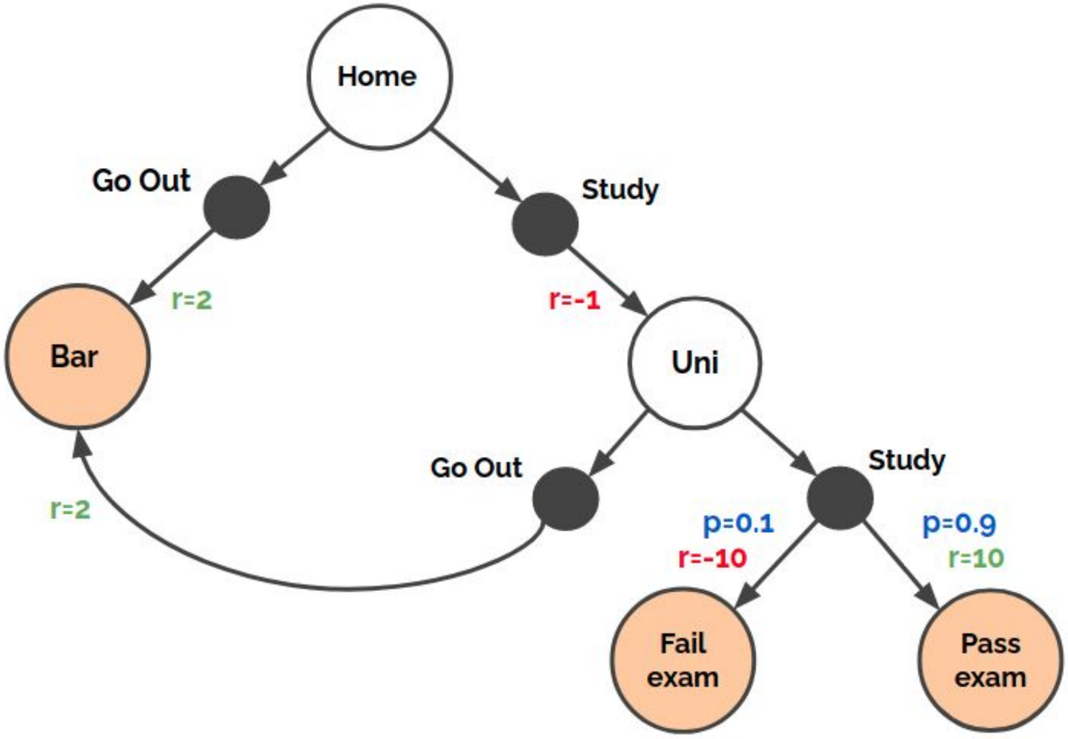
Can write q(s,a) as a function of v(s')

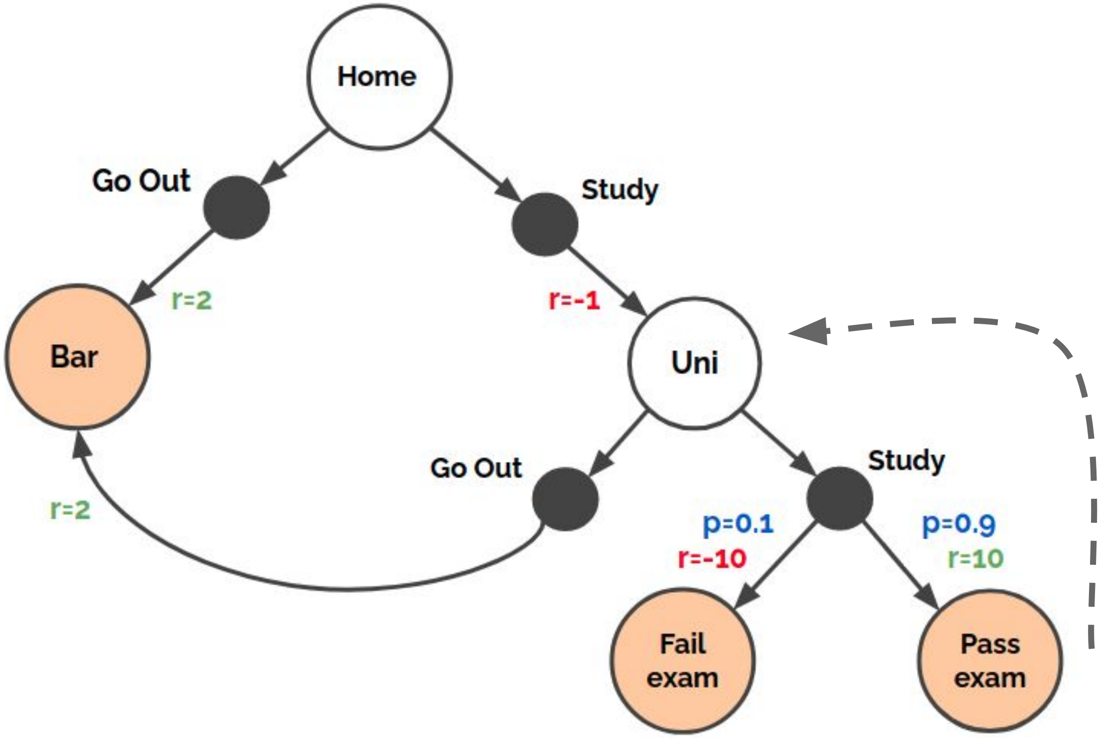
Summary


Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail exam
Pass exam
Can write v(s) as function of q(s,a)
Can write q(s,a) as a function of v(s')

Part Ib

Bellman Equation


Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail exam
Pass exam



Can combine both steps to write v(s) as a function of the next state values v(s')

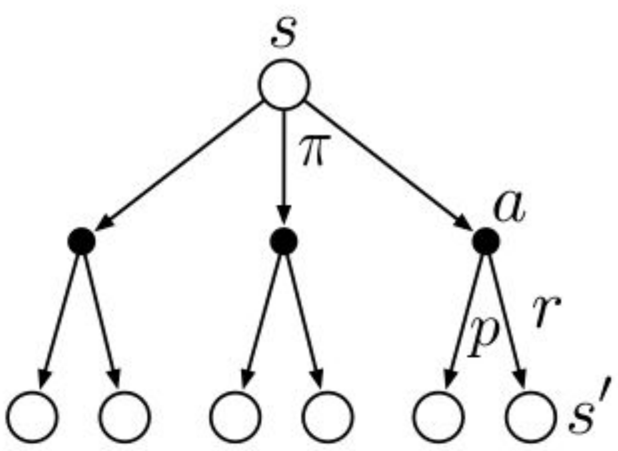
Bellman Equation for v(s)

Bellman Equation for v(s)

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

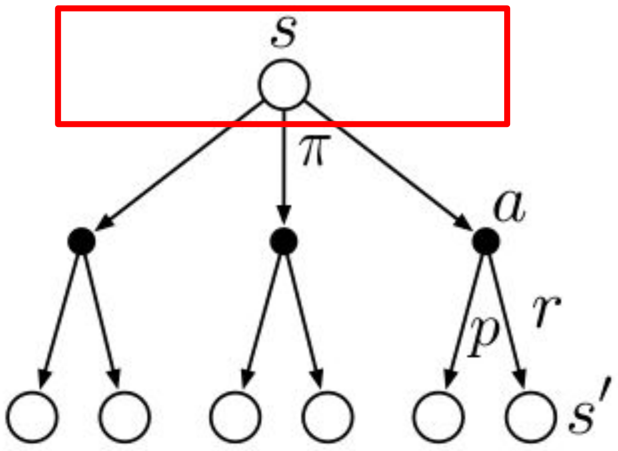
Bellman Equation for v(s)

$$v^\pi(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^\pi(s') \right]$$



Bellman Equation for v(s)

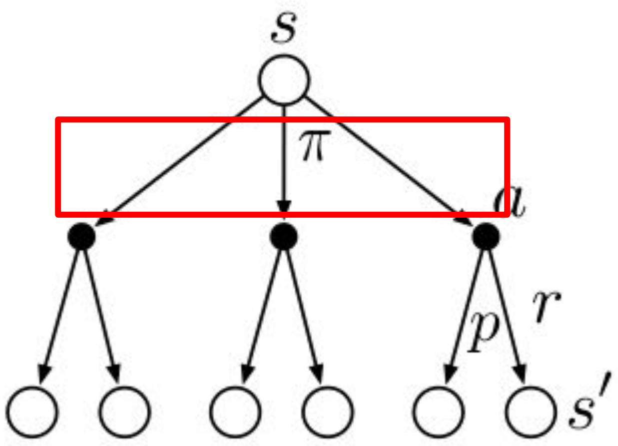
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$



The value of a state is equal to

Bellman Equation for v(s)

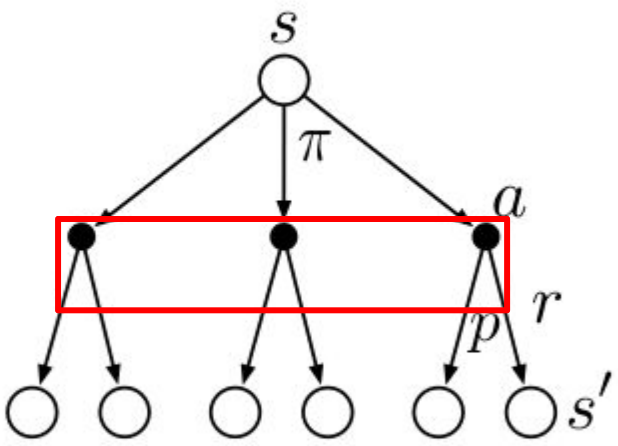
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$



The value of a state is equal to the average over all action probabilities

Bellman Equation for v(s)

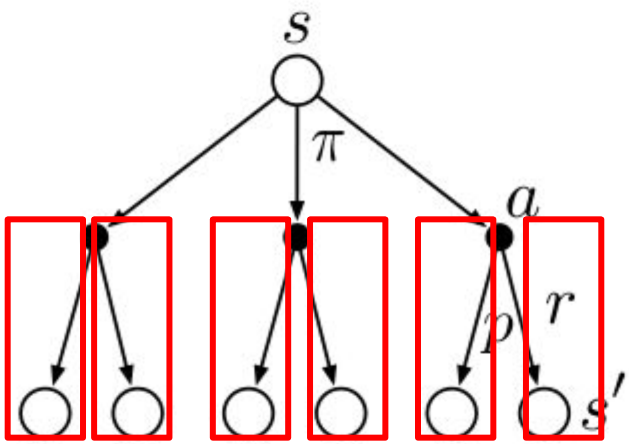
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$



The value of a state is equal to the average over all action probabilities of each average over the resulting transition probabilities

Bellman Equation for v(s)

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$



The value of a state is equal to the average over all action probabilities of each average over the resulting transition probabilities of the reward plus next state value of that transition

Recursive

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\Big[r(s,a,s') + \gamma \cdot v^\pi(s')\Big]$$

Recursive

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s, a, s') + \gamma \cdot v^{\pi}(s') \right]$$

The Bellman Equation is *recursive*

Every state value can be written as a function of the values at states that may follow it

System of equations

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s, a, s') + \gamma \cdot v^{\pi}(s') \Big]$$

System of equations

$$v^\pi(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v^\pi(s') \Big]$$

$v^\pi(s=1)$ = ...

$v^\pi(s=2)$ = ...

$v^\pi(s=3)$ = ...

$v^\pi(s=4)$ = ...

$v^\pi(s=5)$ = ...

System of equations

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

$v^\pi(s=1)$ = ...

$v^\pi(s=2)$ = ...

$v^\pi(s=3)$ = ...

$v^\pi(s=4)$ = ...

$v^\pi(s=5)$ = ...

In each equation you plug in the correct transition probabilities and rewards from that state

System of equations

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\Big[r(s,a,s') + \gamma \cdot v^\pi(s')\Big]$$

$v^\pi(s=1) = \ldots$

$v^\pi(s=2) = \ldots$

$v^\pi(s=3) = \ldots$

$v^\pi(s=4) = \ldots$

$v^\pi(s=5) = \ldots$

- The Bellman Equation specifies a *system of (linear) equations*

System of equations

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\Big[r(s,a,s') + \gamma \cdot v^\pi(s')\Big]$$

$v^\pi(s=1)$ = ...

$v^\pi(s=2)$ = ...

$v^\pi(s=3)$ = ...

$v^\pi(s=4)$ = ...

$v^\pi(s=5)$ = ...

- The Bellman Equation specifies a *system of (linear) equations*
  - We can write out one equation for each state (|S| in total)

System of equations

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

$v^\pi(s=1) = \ldots$

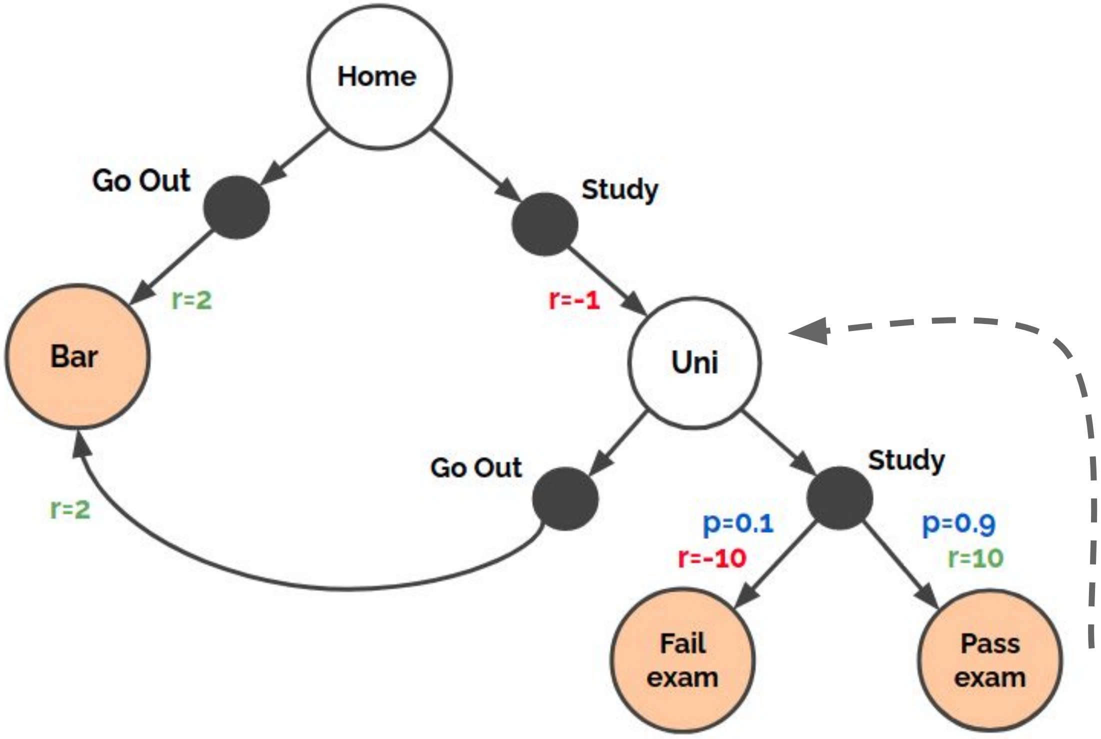
$v^\pi(s=2) = \ldots$

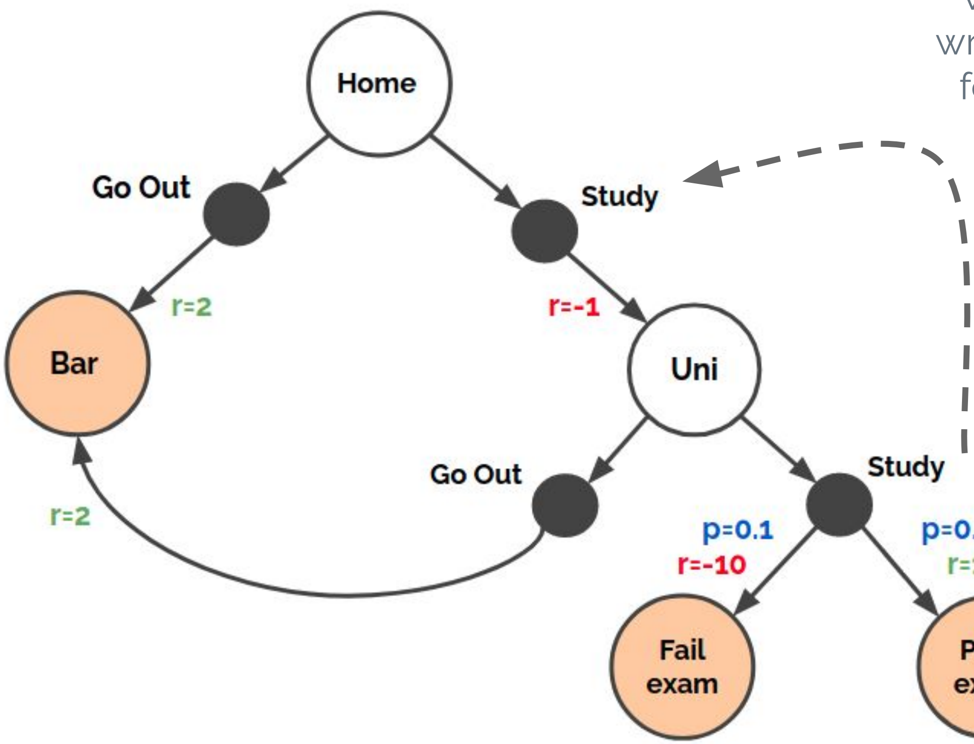
$v^\pi(s=3) = \ldots$

$v^\pi(s=4) = \ldots$

$v^\pi(s=5) = \ldots$

- The Bellman Equation specifies a *system of (linear) equations*
  - We can write out one equation for each state (|S| in total)
  - The v(s) values of each state are the unknowns (|S| unknowns)


Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail
exam
Pass
exam



We can of course also write a Bellman equation for state-action values q(s,a)

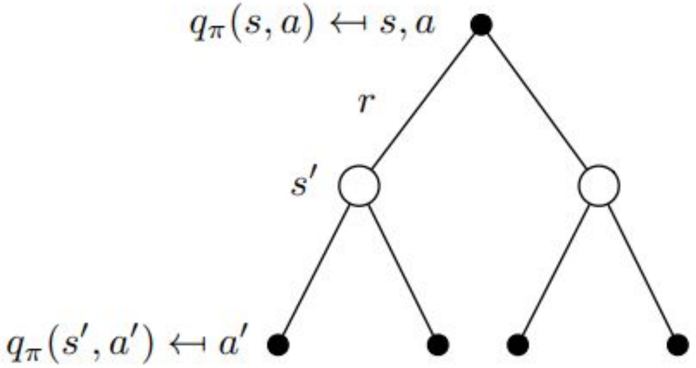
Bellman Equation for q(s,a)

Bellman Equation for q(s,a)

$$q^\pi(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot \mathbb{E}_{a' \sim \pi(a'|s')}[q^\pi(s',a')]\Big]$$

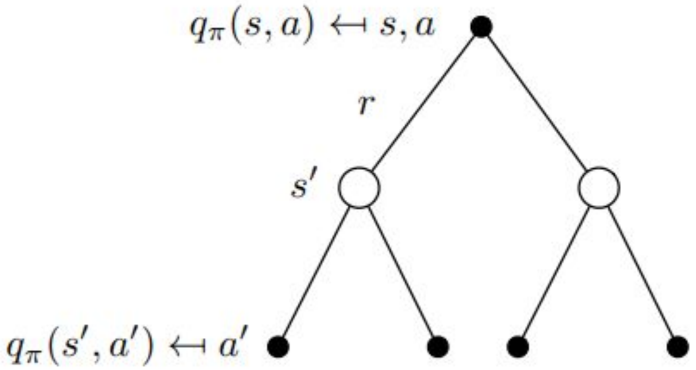
Bellman Equation for q(s,a)

$$q^\pi(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot \mathbb{E}_{a' \sim \pi(a'|s')}[q^\pi(s',a')]\Big]$$



Bellman Equation for q(s,a)

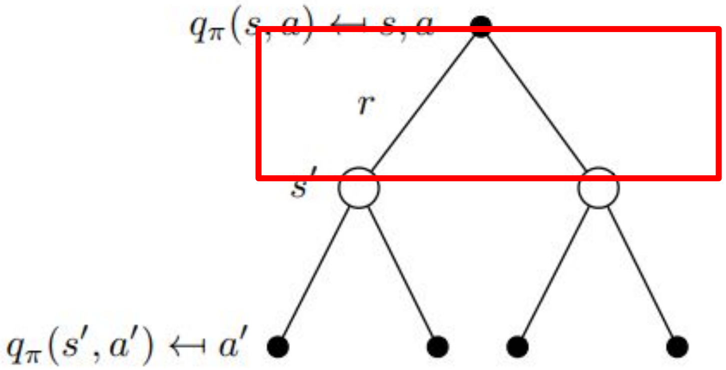
$$q^\pi(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \mathbb{E}_{a' \sim \pi(a'|s')}[q^\pi(s',a')]\right]$$



Same equation

Bellman Equation for q(s,a)

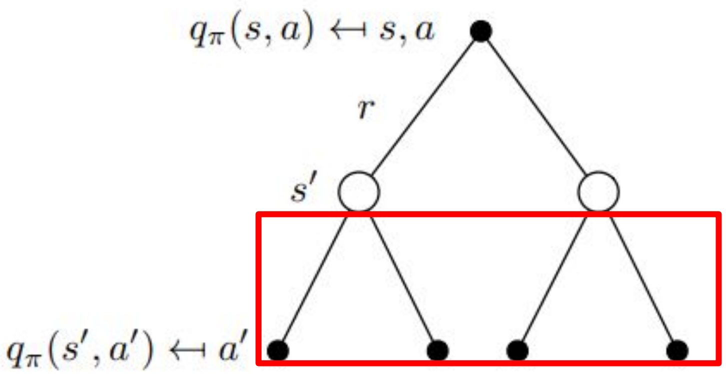
$$q^{\pi}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \mathbb{E}_{a' \sim \pi(a'|s')}[q^{\pi}(s',a')]\right]$$



Same equation, but we now first sum over transition probabilities

Bellman Equation for q(s,a)

$$q^\pi(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \, \mathbb{E}_{a' \sim \pi(a'|s')}[q^\pi(s',a')]\right]$$



Same equation, but we now first sum over transition probabilities, and then over the action probabilities

Bellman Equation from building blocks

Bellman Equation from building blocks

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)}\Big[q^{\pi}(s,a)\Big]$$

v(s) from q(s,a)

Bellman Equation from building blocks

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\Big[q^\pi(s,a)\Big]$$

v(s) from q(s,a)

$$q^\pi(s,a) = \mathbb{E}_{s'\sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot v^\pi(s)\Big]$$

q(s,a) from v(s)

Bellman Equation from building blocks

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\Big[q^\pi(s,a)\Big]$$

$$q^\pi(s,a) = \mathbb{E}_{s'\sim p(s'|s,a)}\Big[r(s,a,s') + \gamma\cdot v^\pi(s)\Big]$$

Bellman Equation from building blocks

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)}\Big[q^{\pi}(s,a)\Big]$$

$$q^{\pi}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot v^{\pi}(s)\Big]$$

Substitute q(s,a) to get the Bellman Equation for state values v(s)

Bellman Equation from building blocks

Substitute v(s) to get the Bellman Equation for state-action values q(s,a)

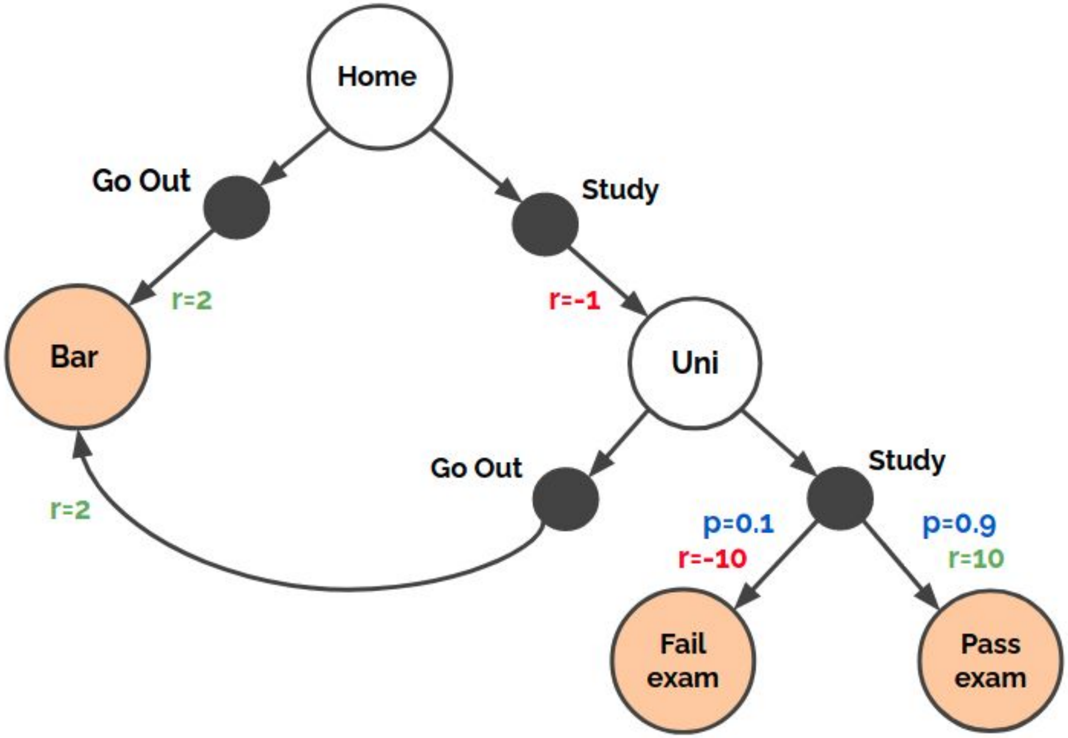
$$v^\pi(s) = \mathbb{E}_{a \sim \pi(a|s)}\Big[q^\pi(s,a)\Big]$$

$$q^\pi(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot v^\pi(s)\Big]$$

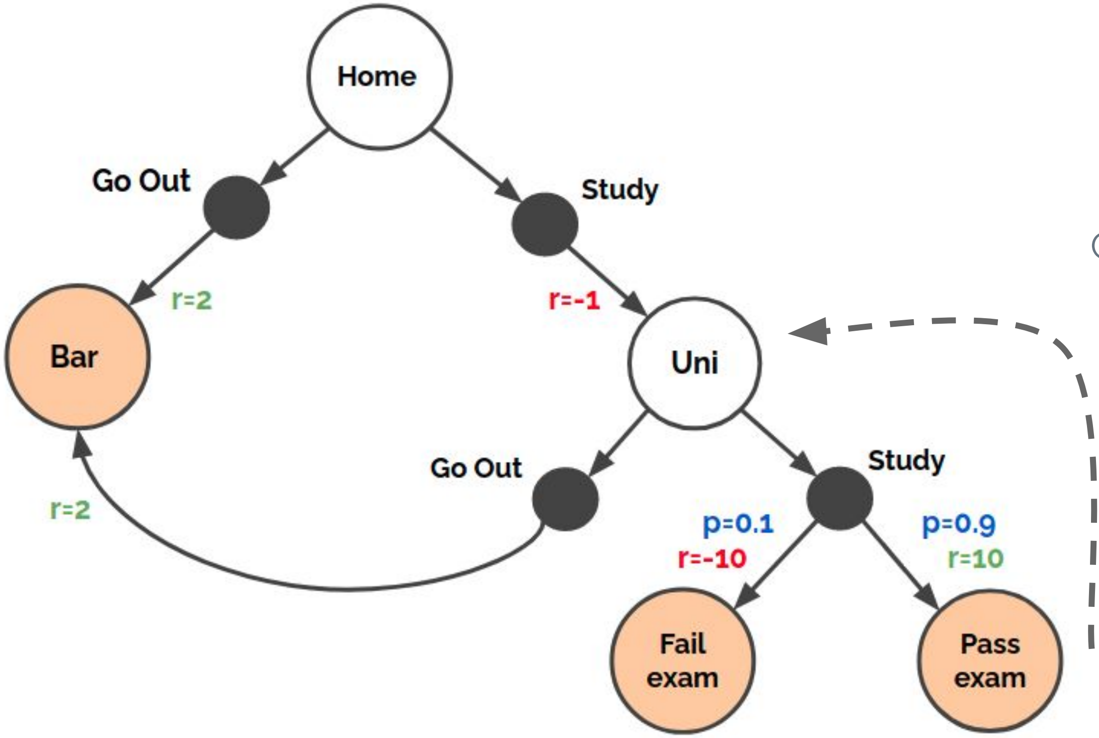
Substitute q(s,a) to get the Bellman Equation for state values v(s)

Write this out yourself at home!

Summary

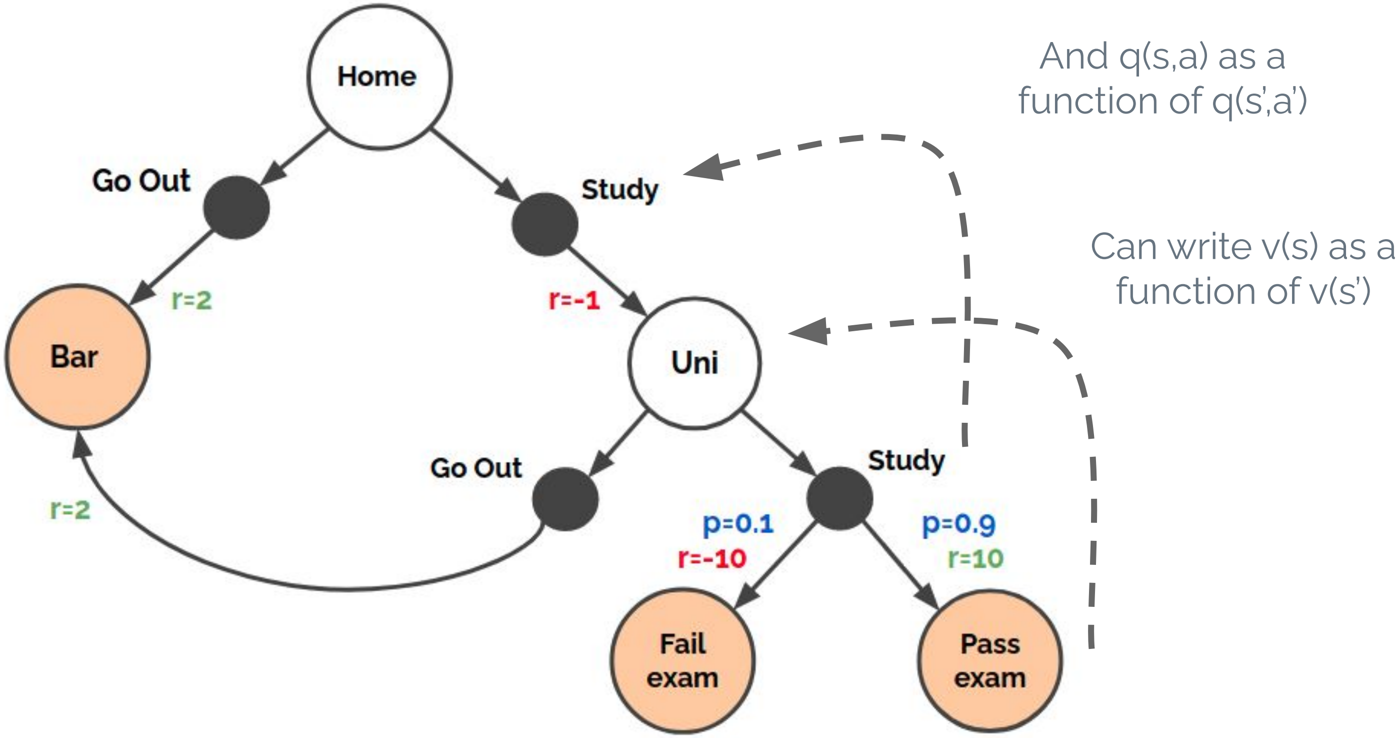

Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail exam
Pass exam

Summary



Can write v(s) as a function of v(s')

Summary



Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail exam
Pass exam
And q(s,a) as a function of q(s',a')
Can write v(s) as a function of v(s')

Part II

Policy Evaluation

Policy Evaluation

Policy Evaluation

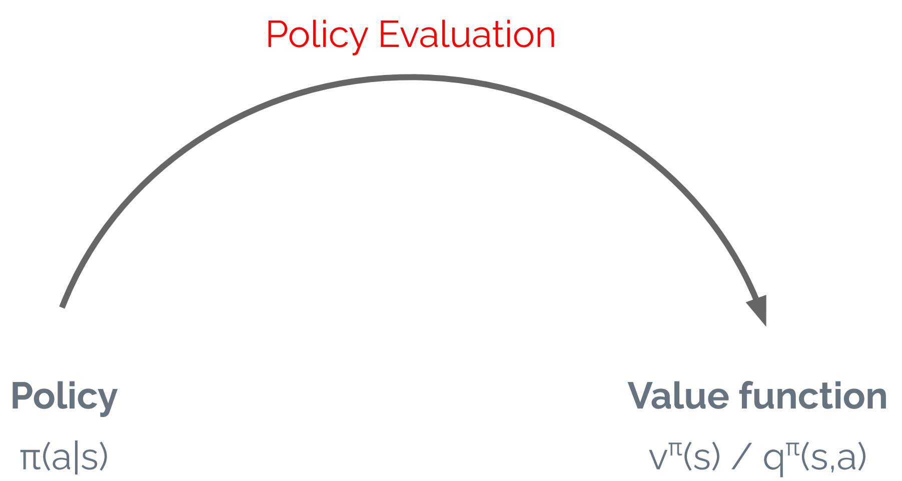
Policy

$\pi(a|s)$

Value function

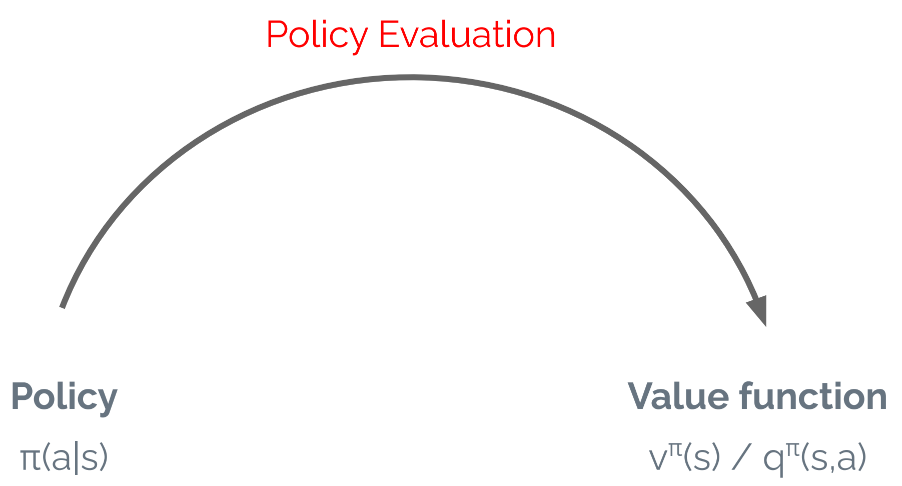
$v^{\pi}(s)$ / $q^{\pi}(s,a)$

Policy Evaluation



Compute the value function of a given policy

Policy Evaluation



Compute the value function of a given policy

We can efficiently compute this through *Dynamic Programming* on the Bellman Equation

Dynamic Programming (DP)

Dynamic Programming (DP)

General concept:

Dynamic Programming (DP)

General concept:

- Break a large problem into smaller subproblems

Dynamic Programming (DP)

General concept:

- Break a large problem into smaller subproblems
- Efficiently store and reuse intermediate results

Dynamic Programming (DP)

General concept:

- Break a large problem into smaller subproblems
- Efficiently store and reuse intermediate results
- Repeatedly solving the small problem solves the big problem

Dynamic Programming (DP)

Dynamic Programming (DP)

In the context of Markov Decision Processes:

Dynamic Programming (DP)

In the context of Markov Decision Processes:

- Small subproblem given by the Bellman Equation

Dynamic Programming (DP)

In the context of Markov Decision Processes:

- Small subproblem given by the Bellman Equation
- Repeatedly solving it gives us the Value Function

Policy Evaluation through DP

Policy Evaluation through DP

Compute the value of a given policy

Policy Evaluation through DP

Compute the value of a given policy

Input: a policy π(a|s), an MDP (p(s'|s,a), r(s,a,s'), γ)

Policy Evaluation through DP

Compute the value of a given policy

Input: a policy $\pi(a|s)$, an MDP ($p(s'|s,a)$, $r(s,a,s')$, γ)

Algorithm:

Policy Evaluation through DP

Compute the value of a given policy

Input: a policy $\pi(a|s)$, an MDP $(p(s'|s,a), r(s,a,s'), \gamma)$

Algorithm:

- Initialize $v(s)=0$ for all s

Policy Evaluation through DP

Compute the value of a given policy

Input: a policy $\pi(a|s)$, an MDP ($p(s'|s,a)$, $r(s,a,s')$, γ)

Algorithm:

- Initialize $v(s)=0$ for all s
- Sweep through all states, updating according to Bellman Equation:

Policy Evaluation through DP

Compute the value of a given policy

Input: a policy π(a|s), an MDP (p(s'|s,a), r(s,a,s'), γ)

Algorithm:

- Initialize v(s)=0 for all s
- Sweep through all states, updating according to Bellman Equation:

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

Policy Evaluation through DP

Compute the value of a given policy

Input: a policy π(a|s), an MDP (p(s'|s,a), r(s,a,s'), γ)

Algorithm:

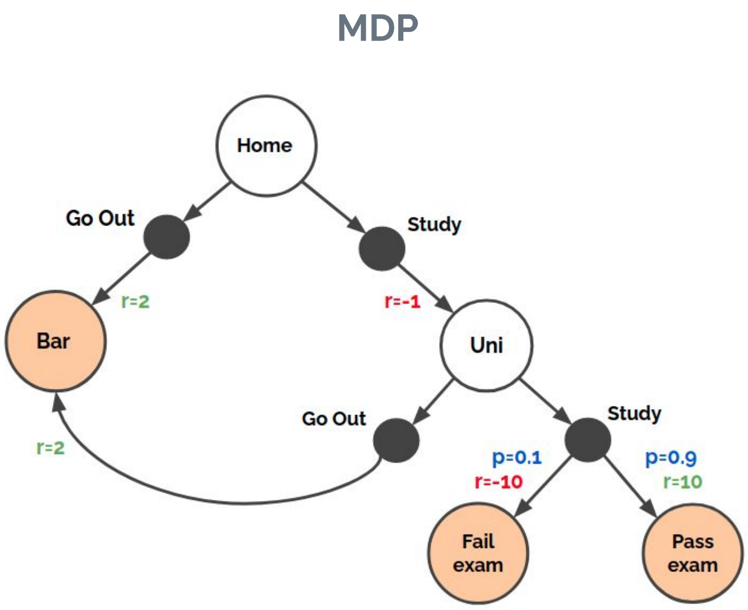
- Initialize v(s)=0 for all s
- Sweep through all states, updating according to Bellman Equation:

$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\Big[r(s,a,s') + \gamma \cdot v^\pi(s')\Big]$$

- Until v(s) converges

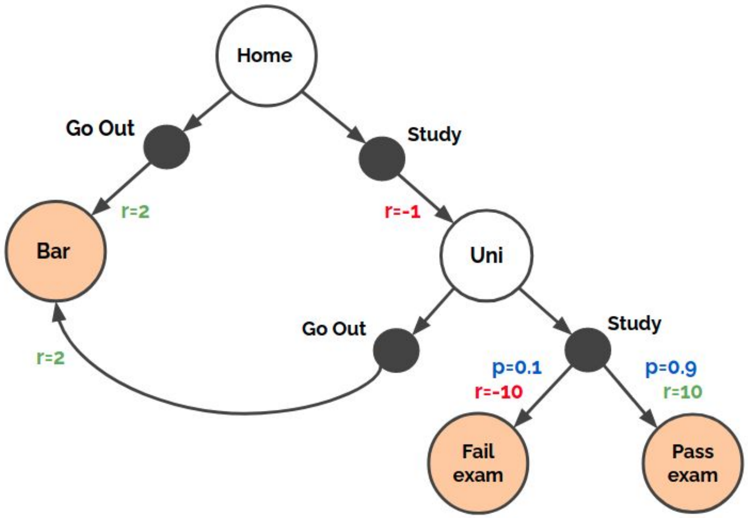
Policy Evaluation through DP: Example

Policy Evaluation through DP: Example



Policy Evaluation through DP: Example

MDP



Bellman Equation

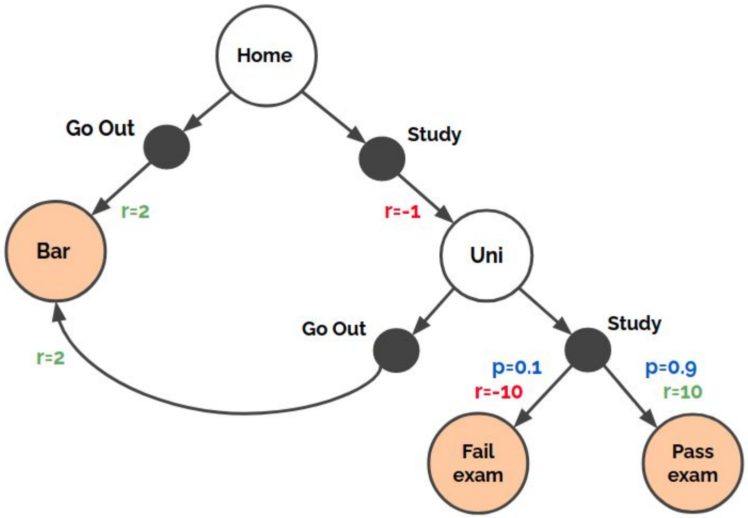
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.0
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Bellman Equation

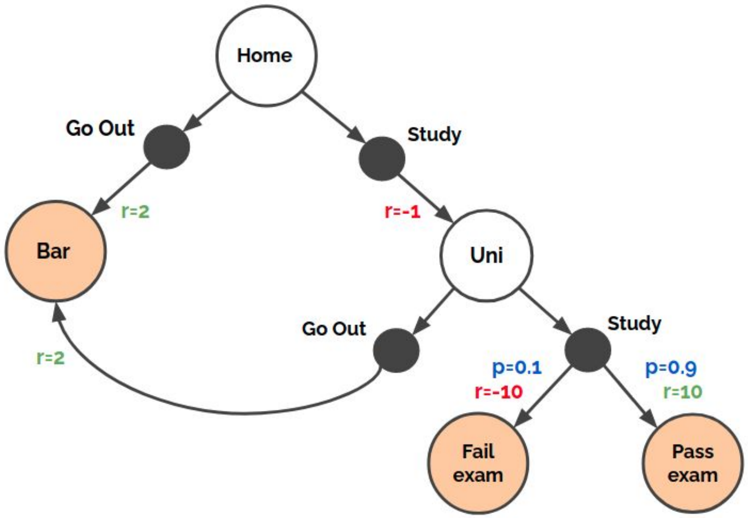
$$v^{\pi}(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^{\pi}(s')\right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.0
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Bellman Equation

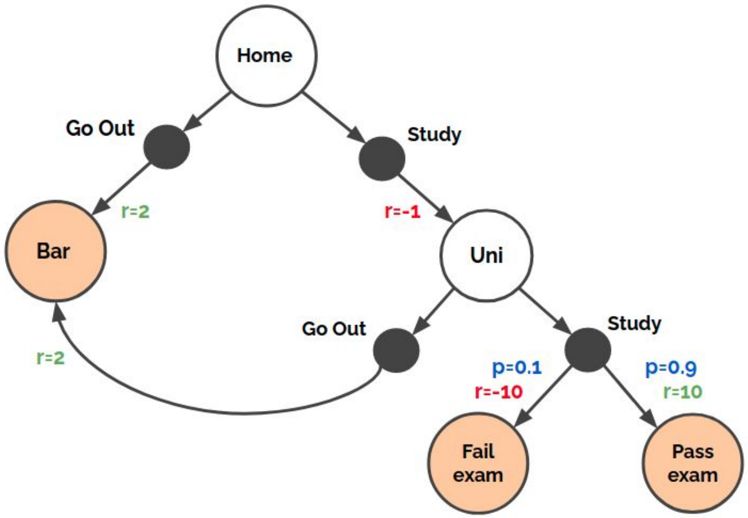
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.0
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Home)?

Bellman Equation

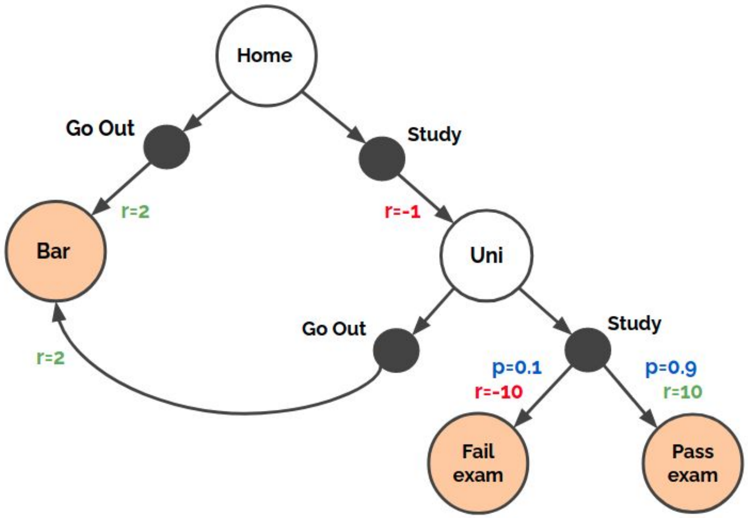
$$v^{\pi}(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\Big[r(s,a,s') + \gamma \cdot v^{\pi}(s')\Big]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.0
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Home)?

A: 0.5 · (2.0 + 0) + 0.5 · (-1.0 + 0.0) = 0.5

Bellman Equation

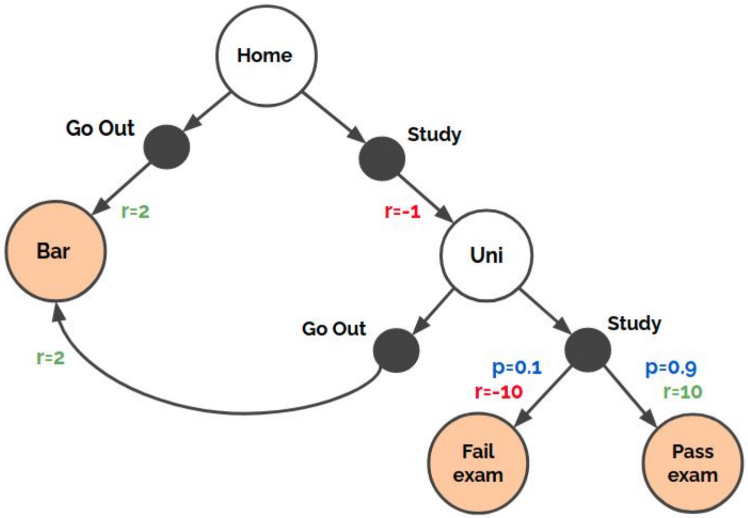
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	**0.5**
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Home)?

A: 0.5 · (2.0 + 0) + 0.5 · (-1.0 + 0.0) = 0.5

Bellman Equation

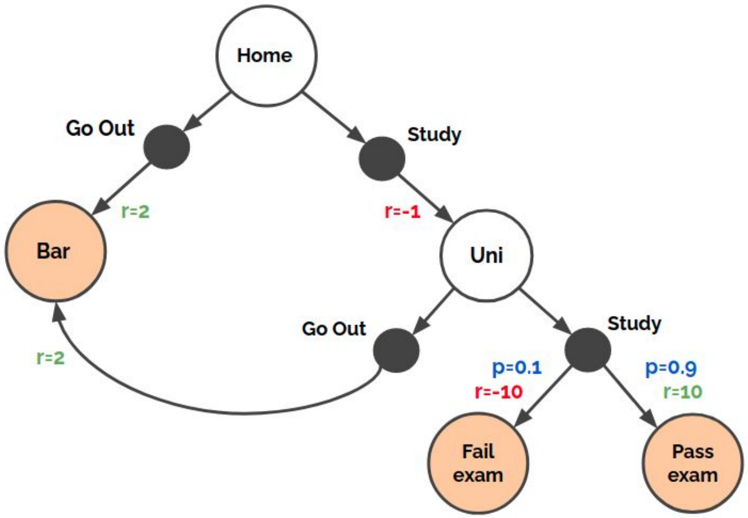
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Bar)?

Bellman Equation

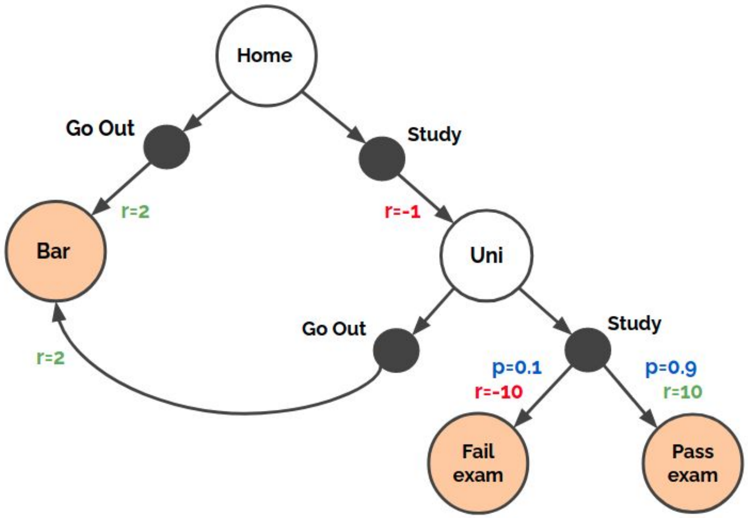
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	**0.0**
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Bar)?

A: 0.0 (terminal)

Bellman Equation

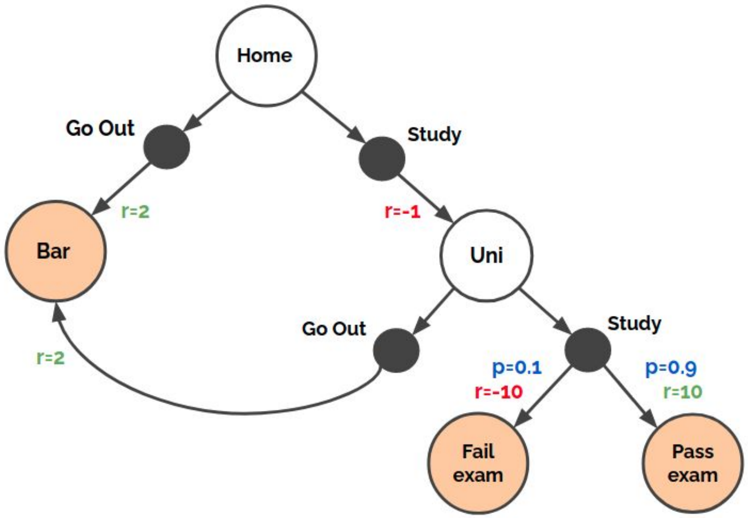
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Uni)?

Bellman Equation

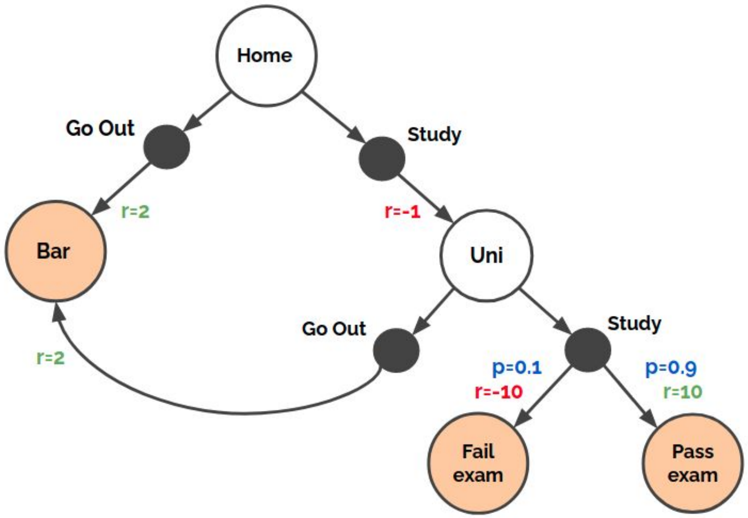
$$v^{\pi}(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^{\pi}(s')\right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	0.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Uni)?

A: 0.5 · (2.0 + 0) +
0.5 · (0.1 · (-10 + 0.0) + 0.9 · (10 + 0))
= 5.0

Bellman Equation

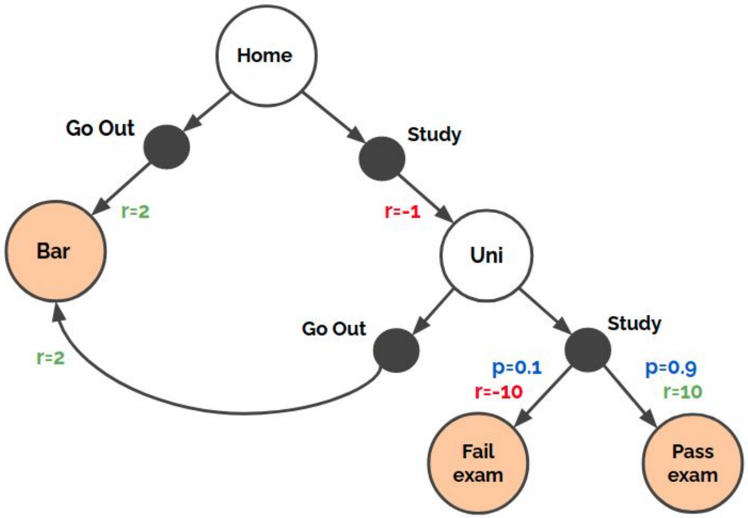
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	**5.0**
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Uni)?

A: $0.5 \cdot (2.0 + 0) +$
$0.5 \cdot (0.1 \cdot (-10 + 0.0) + 0.9 \cdot (10 + 0))$
$= 5.0$

Bellman Equation

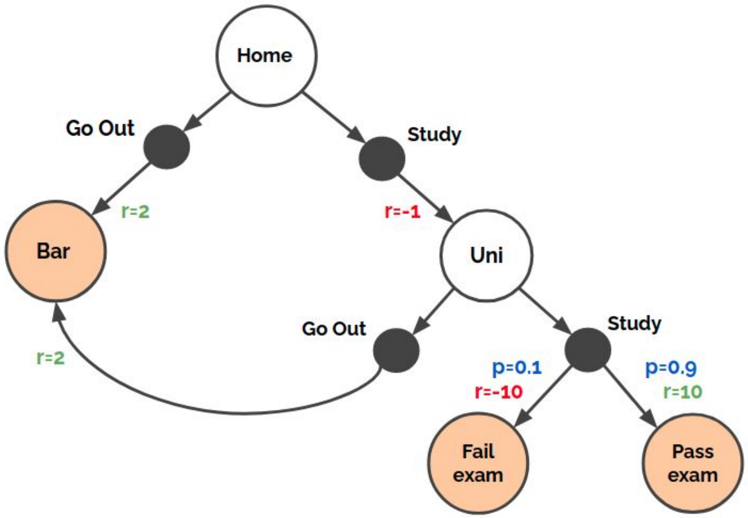
$$v^\pi(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^\pi(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	5.0
Fail exam	**0.0**
Pass exam	0.0

MDP



Fail exam is terminal

Bellman Equation

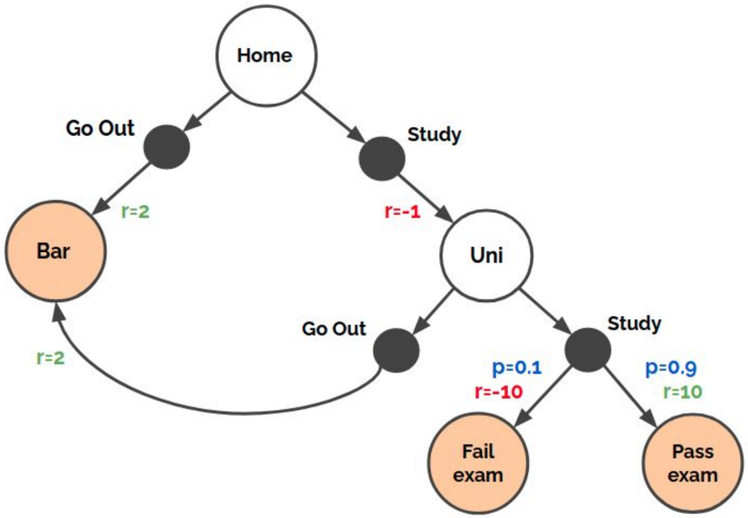
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	5.0
Fail exam	0.0
Pass exam	**0.0**

MDP



Pass exam is terminal

Bellman Equation

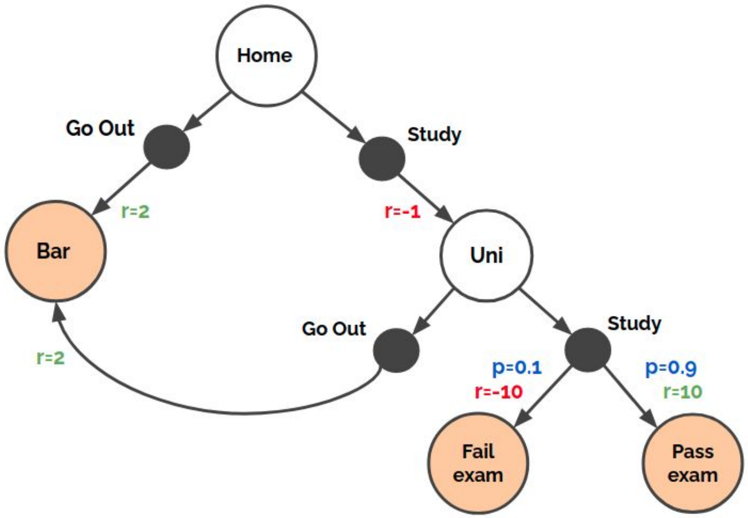
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	5.0
Fail exam	0.0
Pass exam	0.0

MDP



Repeat!

Bellman Equation

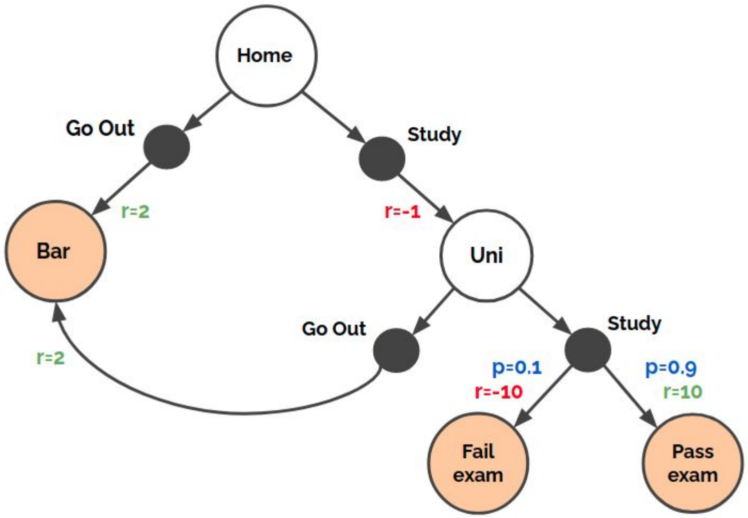
$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	5.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Home)?

A:

Bellman Equation

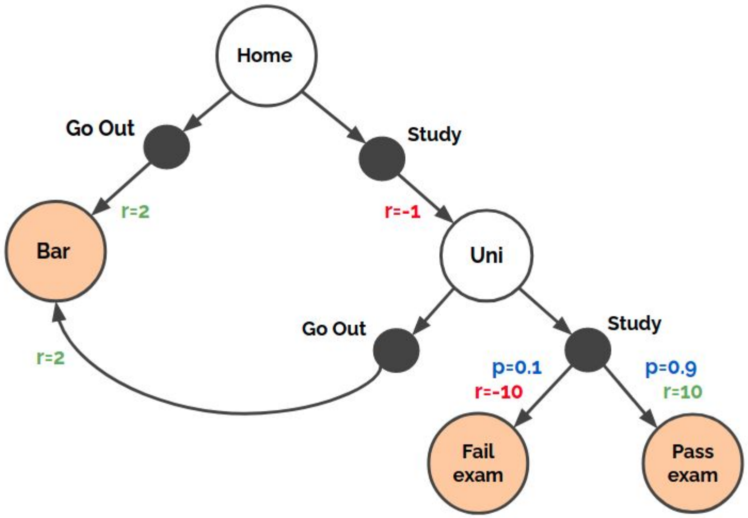
$$v^{\pi}(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\Big[r(s,a,s') + \gamma \cdot v^{\pi}(s')\Big]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	5.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Home)?

A: 0.5 · (2.0 + 0) + 0.5 · (-1.0 + **5.0**) = 3.0

Bellman Equation

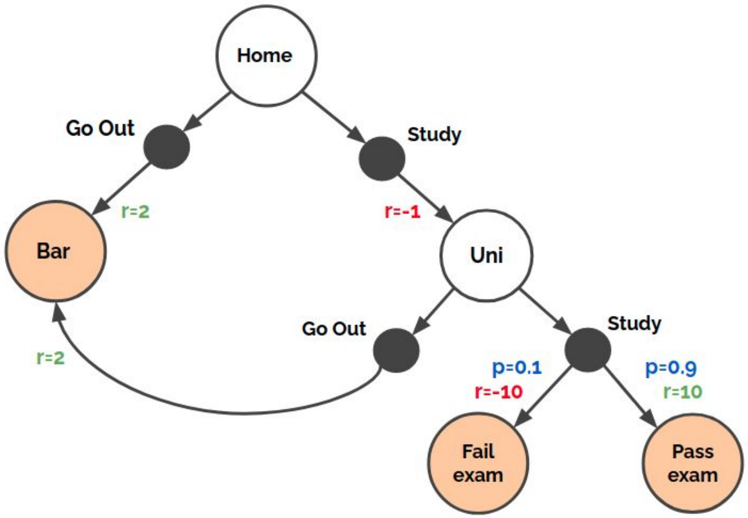
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	**3.0**
Bar	0.0
Uni	5.0
Fail exam	0.0
Pass exam	0.0

MDP



Assume random policy

Q: What is the update of v(Home)?

A: $0.5 \cdot (2.0 + 0) + 0.5 \cdot (-1.0 + 5.0) = 3.0$

Bellman Equation

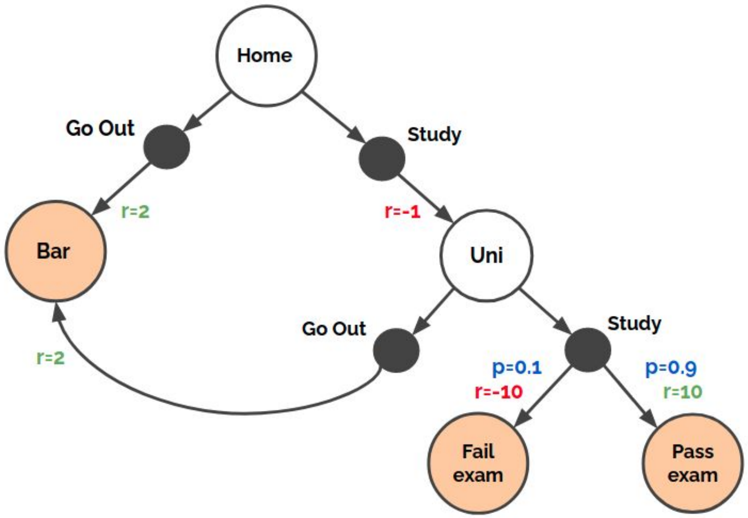
$$v^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\mathbb{E}_{s'\sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\pi(s')\right]$$

Policy Evaluation through DP: Example

Solution table

s	v(s)
Home	0.5
Bar	0.0
Uni	5.0
Fail exam	0.0
Pass exam	0.0

MDP



Repeat until convergence

(i.e. v(s) estimates stabilize)

= result is true $v^{\pi}(s)$

Bellman Equation

$$v^{\pi}(s) = \mathbb{E}_{a \sim \pi(a|s)} \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^{\pi}(s') \right]$$

Summary

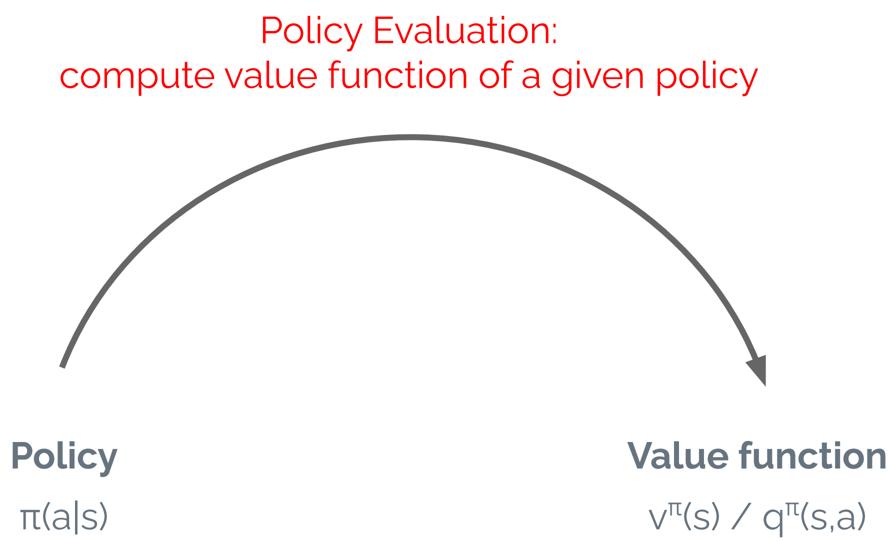
Policy

$\pi(a|s)$

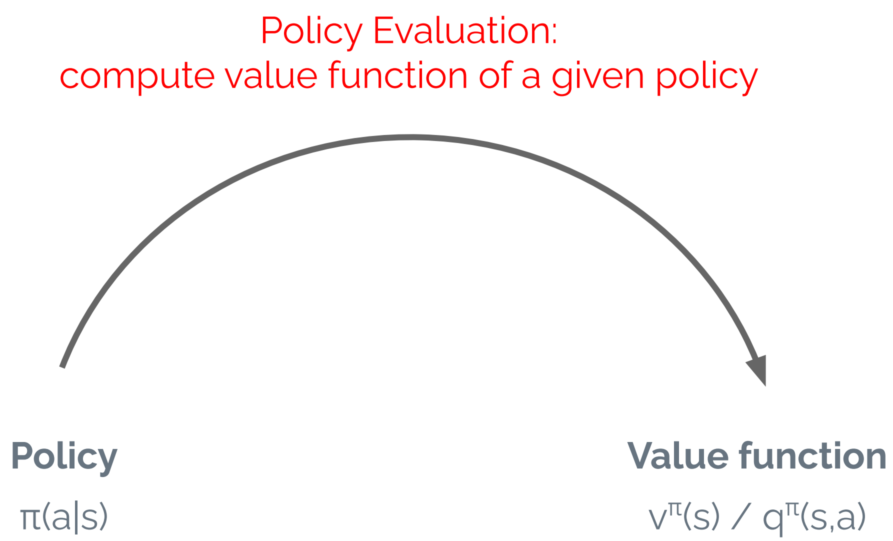
Value function

$v^{\pi}(s)$ / $q^{\pi}(s,a)$

Summary


Policy Evaluation:
compute value function of a given policy
Policy
$\pi(a|s)$
Value function
$v^{\pi}(s)$ / $q^{\pi}(s,a)$

Summary



We can efficiently compute the value of a given policy through Dynamic Programming,

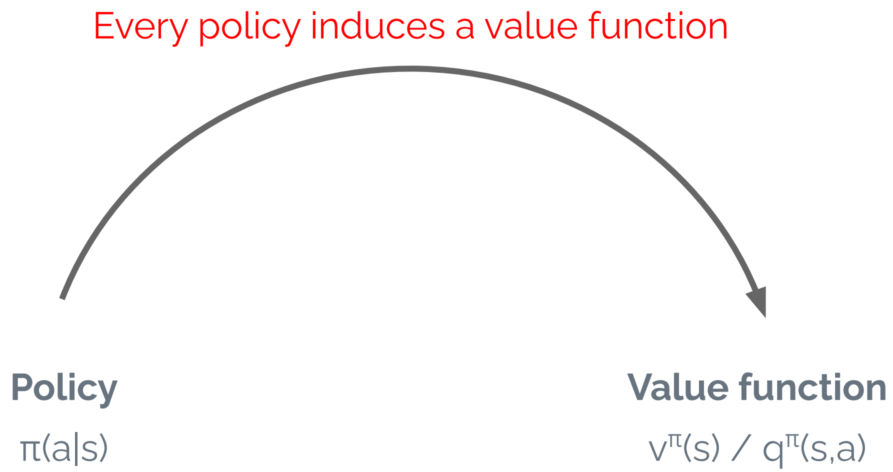
repeatedly solving the Bellman Equation

Part III

Implicit Policies

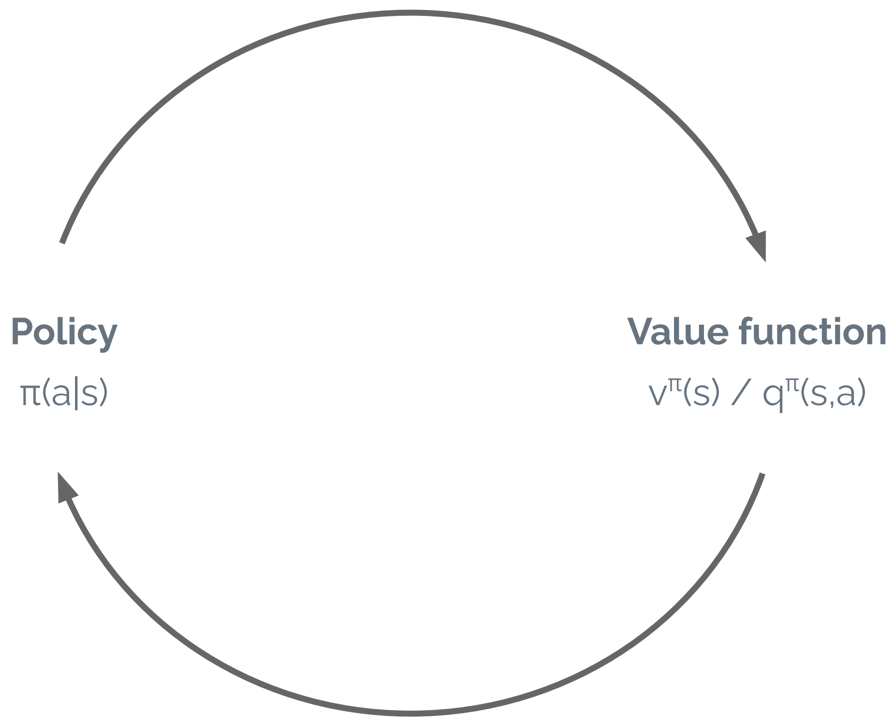
From v(s) / q(s,a) to new π

From v(s) / q(s,a) to new π



From v(s) / q(s,a) to new π

Every policy induces a value function



Can we also use a given value function to define a new policy?

Explicit policy

Explicit policy

Directly store the policy probabilities

Explicit policy

Directly store the policy probabilities

s	**a** Go out	Study
Home	0.5	0.5
Uni	0.5	0.5
Bar	-	-
Pass exam	-	-
Fail exam	-	-

Implicit policy

Implicit policy

Only store value function, define policy as function of the value estimates

Implicit policy

Only store value function, define policy as function of the value estimates

s	$q^{\text{random}}(s, a)$	
	Go out	Study
Home	2.0	4.0
Uni	2.0	8.0
Bar	0.0	0.0
Pass exam	0.0	0.0
Fail exam	0.0	0.0

Implicit policy

Only store value function, define policy as **function f**() of the value estimates

s	$q^{\text{random}}(s,a)$ Go out	Study
Home	2.0	4.0
Uni	2.0	8.0
Bar	0.0	0.0
Pass exam	0.0	0.0
Fail exam	0.0	0.0

→ $\pi(a|s)$ = **f**(v) or **f**(q)

Implicit policy

Only store value function, define policy as function of the value estimates

s	$q^{\text{random}}(s,a)$ Go out	Study
Home	2.0	4.0
Uni	2.0	8.0
Bar	0.0	0.0
Pass exam	0.0	0.0
Fail exam	0.0	0.0

→ $\pi(a|s)$ = **f(**q/v**)**

f() can take many forms:

- greedy (DP)
- Ɛ-greedy (RL)
- Boltzmann (RL)
- etc.

Implicit policy

Only store value function, define policy as function of the value estimates

s	$q^{\text{random}}(s,a)$ Go out	Study
Home	2.0	4.0
Uni	2.0	8.0
Bar	0.0	0.0
Pass exam	0.0	0.0
Fail exam	0.0	0.0

$\pi(a|s) = \mathbf{f}(q/v)$

f() can take many forms:

- **greedy** **(DP)**
- ε-greedy (RL)
- Boltzmann (RL)
- etc.

Example: Greedy policy

Example: Greedy policy

‘Always select the action with the highest state-action value estimate’

Example: Greedy policy

‘Always select the action with the highest state-action value estimate’

For q(s,a):

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

s	$q^{\text{random}}(s,a)$ Go out	Study
Home	2.0	4.0
Uni	2.0	8.0
Bar	0.0	0.0
Pass exam	0.0	0.0
Fail exam	0.0	0.0

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s, a)$$

s	$q^{\text{random}}(s,a)$ Go out	Study
Home	2.0	4.0
Uni	2.0	8.0
Bar	0.0	0.0
Pass exam	0.0	0.0
Fail exam	0.0	0.0

Question: What is $\pi^{\text{greedy}}(\text{Uni})$?

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

s	$q^{\text{random}}(s,a)$ Go out	Study
Home	2.0	4.0
Uni	2.0	8.0
Bar	0.0	0.0
Pass exam	0.0	0.0
Fail exam	0.0	0.0

Question: What is $\pi^{\text{greedy}}(\text{Uni})$?

Answer: Study

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

For v(s):

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

For v(s): More complicated, for each action need to go over the dynamics again

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

For v(s): More complicated, for each action need to go over the dynamics again

$$\pi^{\text{greedy}}(s) = \arg\max_a \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v(s') \Big]$$

Example: Greedy policy

'Always select the action with the highest state-action value estimate'

For q(s,a): Easy to compute the greedy policy (main benefit of state-action values)

$$\pi^{\text{greedy}}(s) = \arg\max_a q(s,a)$$

For v(s): More complicated, for each action need to go over the dynamics again (downside of state values - less useful for action selection)

$$\pi^{\text{greedy}}(s) = \arg\max_a \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v(s') \Big]$$

Policy Improvement

The greedy/max policy is a form of *policy improvement*

Policy Improvement

The greedy/max policy is a form of *policy improvement*

If we take a policy π

Policy Improvement

The greedy/max policy is a form of *policy improvement*

If we take a policy **π**

$\downarrow$

compute its true value function **$q^{\pi}(s,a)$**

Policy Improvement

The greedy/max policy is a form of *policy improvement*

If we take a policy $\boldsymbol{\pi}$

↓

compute its true value function $\mathbf{q^{\pi}(s,a)}$

↓

and then compute a greedy new policy $\boldsymbol{\pi}^{\mathbf{new}} = \text{greedy}(\mathbf{q^{\pi}(s,a)})$

Policy Improvement

The greedy/max policy is a form of *policy improvement*

If we take a policy — $\boldsymbol{\pi}$

compute its true value function — $\mathbf{q}^{\pi}\mathbf{(s,a)}$

and then compute a greedy new policy — $\boldsymbol{\pi}^{\mathbf{new}}$ = greedy($\mathbf{q}^{\pi}\mathbf{(s,a)}$)

Then π^{new} is guaranteed to be a better policy then π

Summary

Summary

Every policy induces a value function

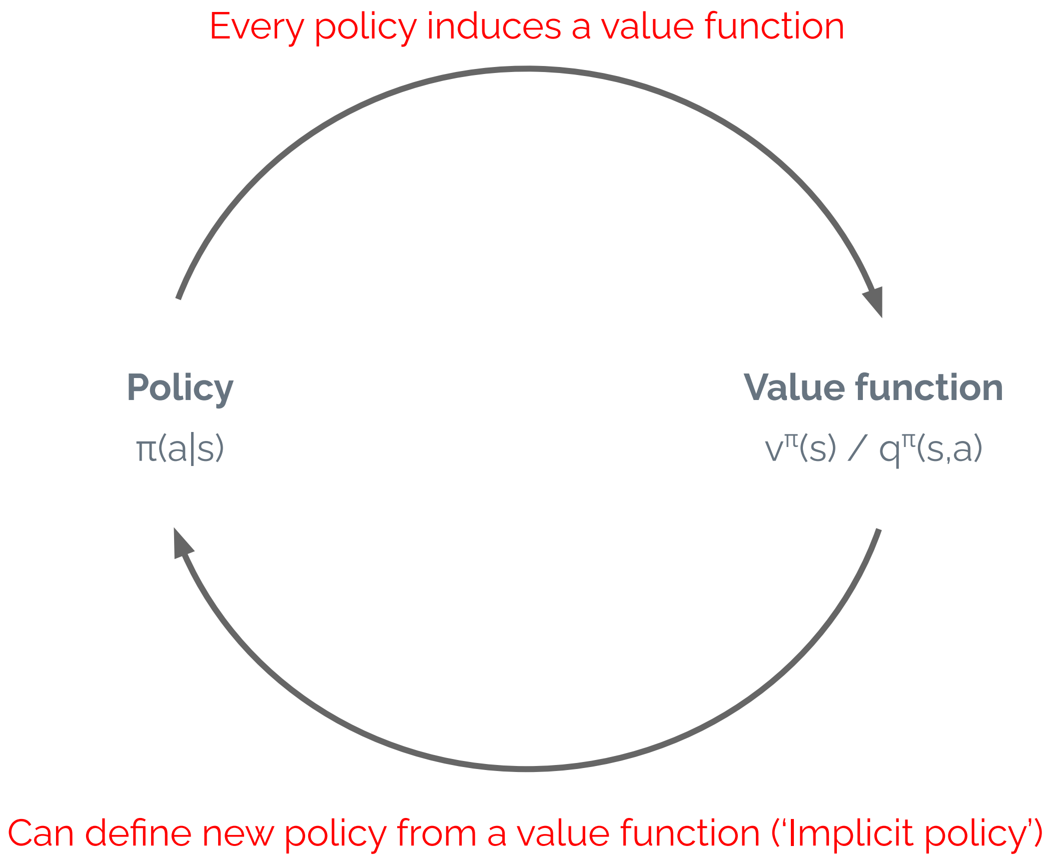
Policy

$\pi(a|s)$

Value function

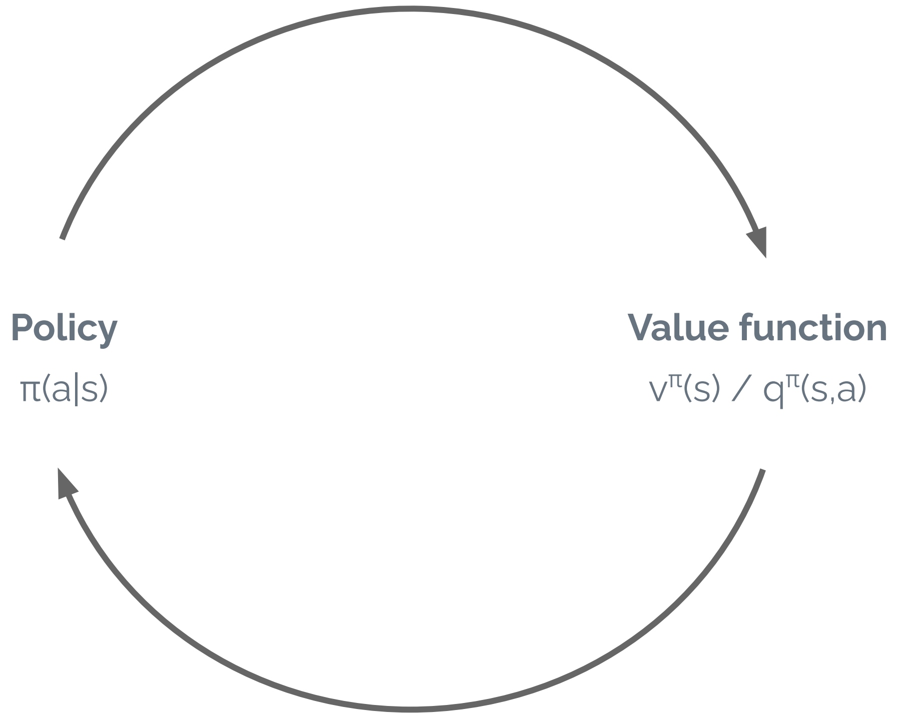
$v^{\pi}(s)$ / $q^{\pi}(s,a)$

Summary


Every policy induces a value function
Policy
π(a|s)
Value function
vπ(s) / qπ(s,a)
Can define new policy from a value function ('Implicit policy')

Summary

Every policy induces a value function



Can define new policy from a value function ('Implicit policy')

The greedy policy over a true value function always improves ('Policy Improvement')

Part IV

Finding v^*, q^* and π^*

Part IVa

Bellman Optimality Equation

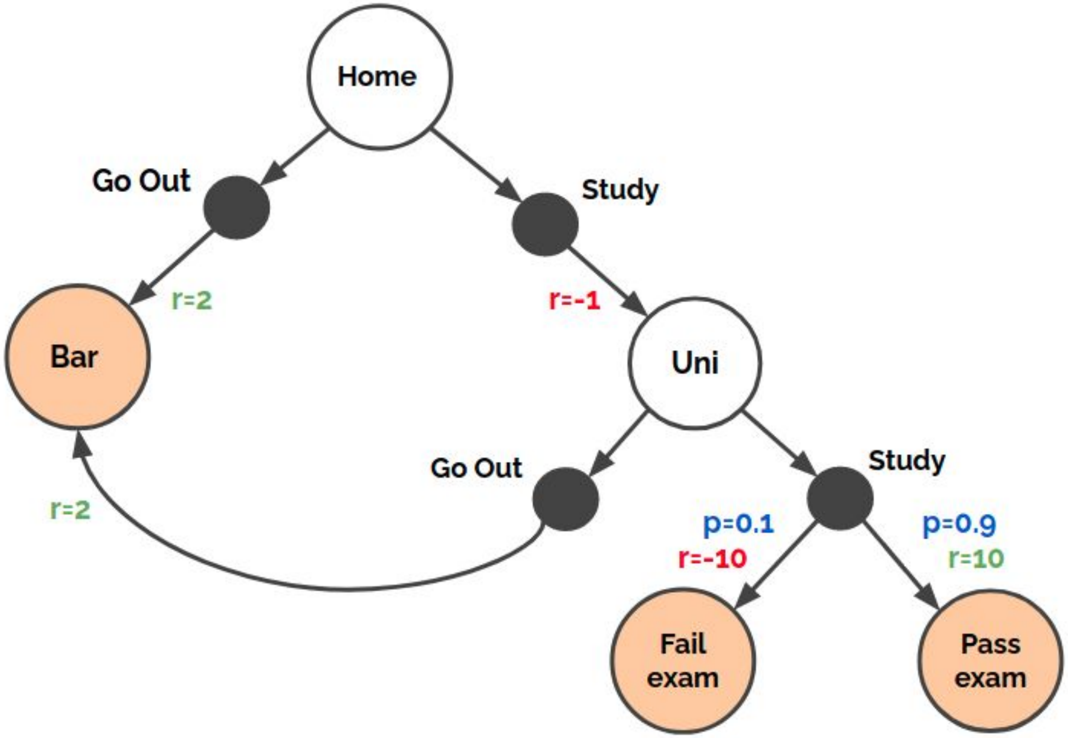
We are of course most interested in the optimal value function $v^*(s)/q^*(s,a)$

and associated optimal policy $\pi^*(a|s)$

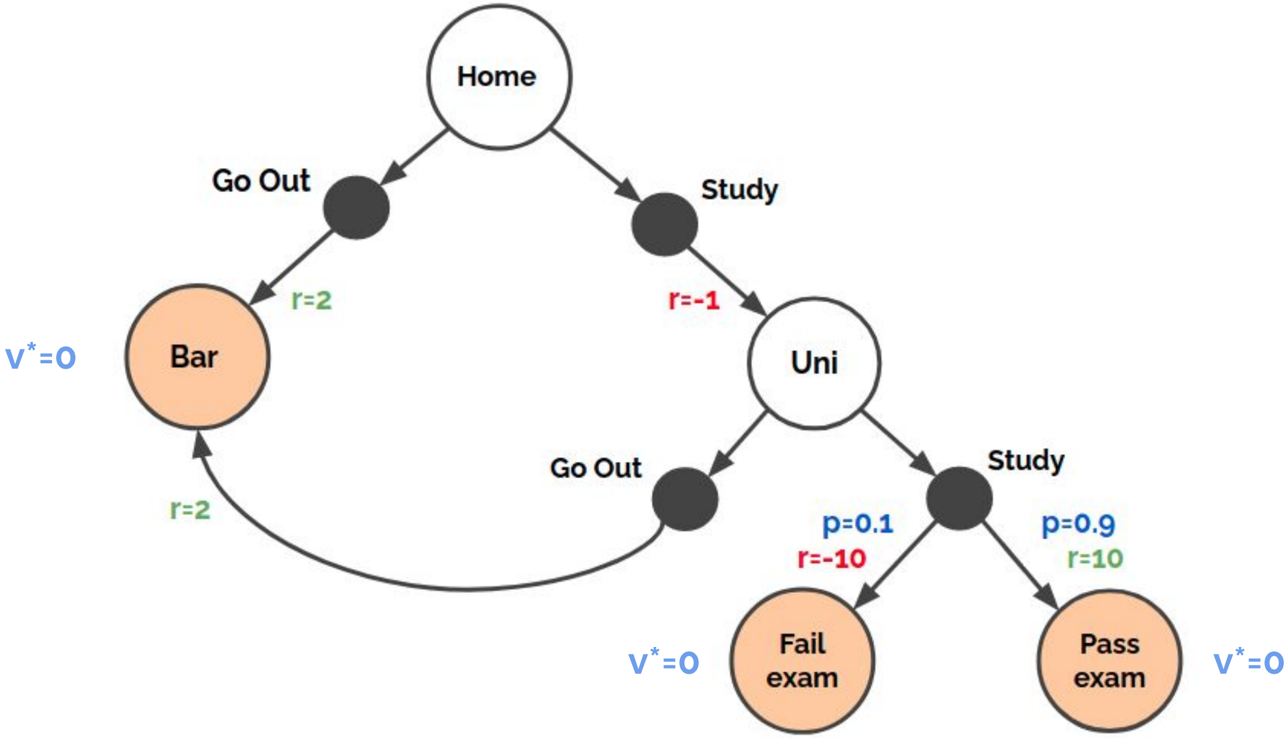
We are of course most interested in the optimal value function $v^*(s)/q^*(s,a)$

and associated optimal policy $\pi^*(a|s)$

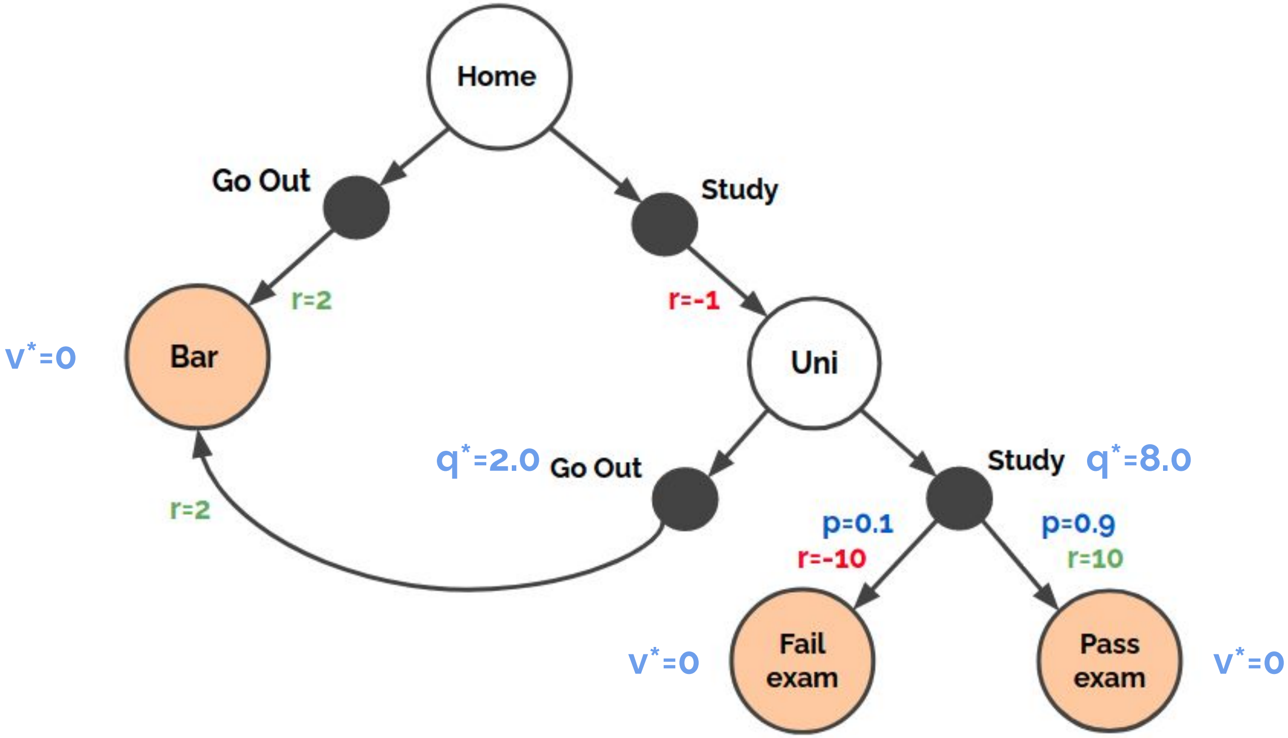
But what changes to our back-up equations in this case?



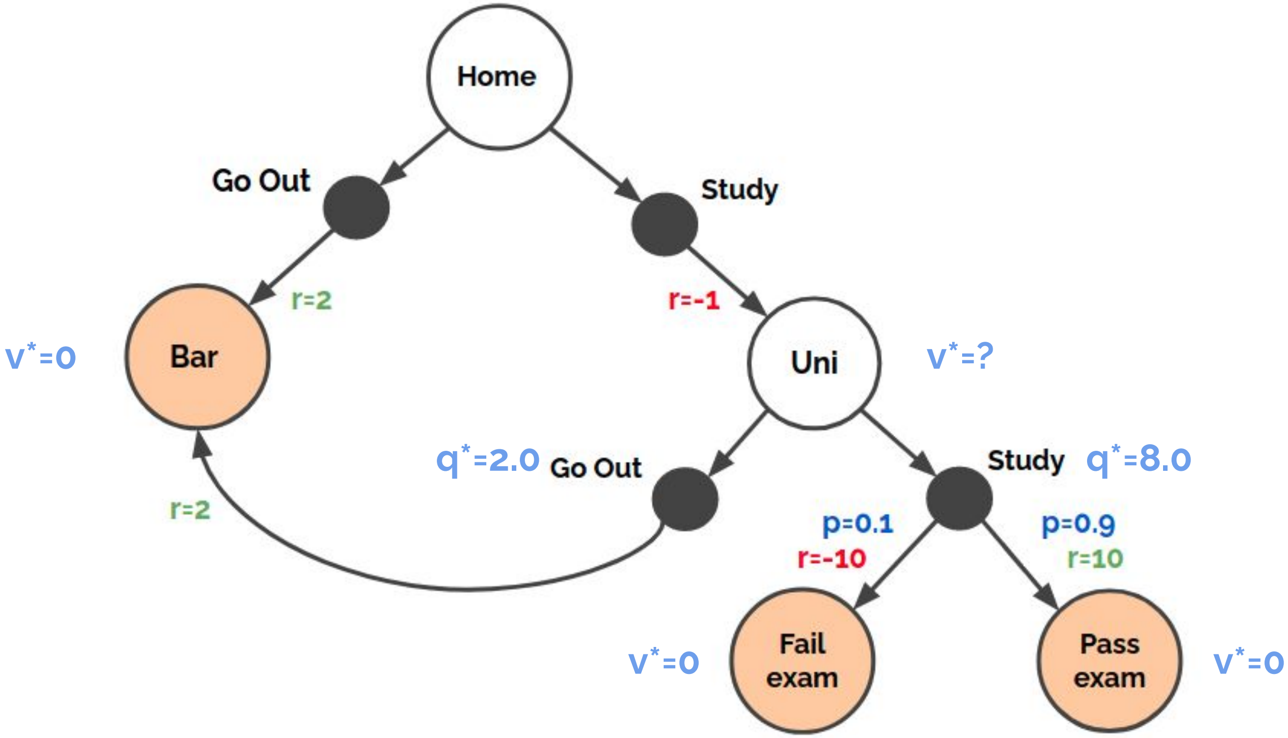
Home
Go Out
Study
r=2
r=-1
Bar
Uni
Go Out
Study
r=2
p=0.1
r=-10
p=0.9
r=10
Fail exam
Pass exam



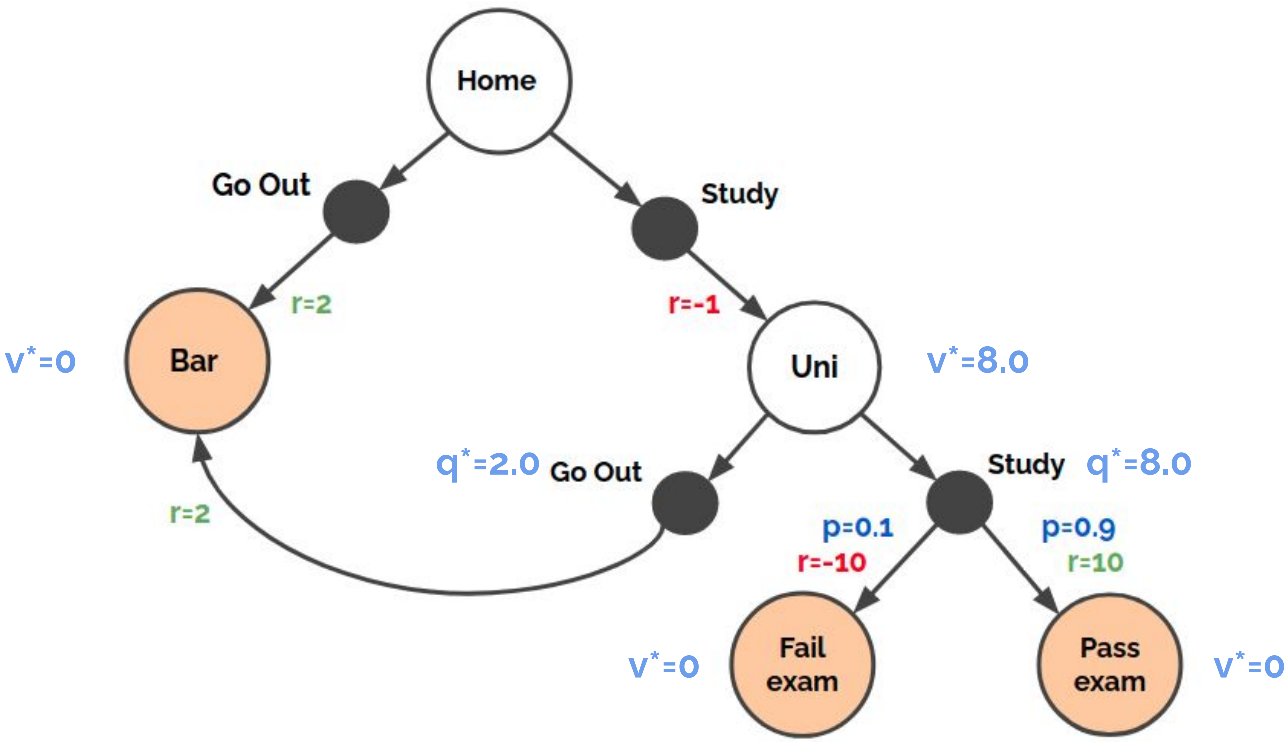
Terminal states values are still 0.0 under the optimal policy



To compute $q^*(s,a)$ from $v^*(s)$ we still average over the transition dynamics



Question: But what is v*(Uni)? (i.e., how do we go from q* to v* under the optimal policy)



Question: But what is v^*(Uni)? (i.e., how do we go from q^* to v^* under the optimal policy)

Answer: v^*(Uni) = 8.0

Under the optimal policy we greedily choose the action with the highest value!

Optimal policy

Optimal policy

Key insight:

The optimal policy is a greedy/max policy with respect to the optimal state-action values

('rational agent')

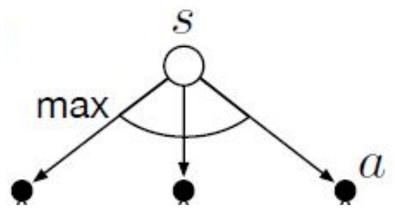
Optimal policy

<u>Key insight:</u>

The optimal policy is a greedy/max policy with respect to the optimal state-action values

('rational agent')

$$v^\star(s) = \max_a q^\star(s,a)$$



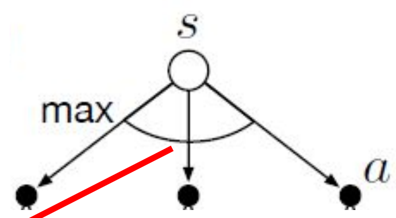
Optimal policy

Key insight:

The optimal policy is a greedy/max policy with respect to the optimal state-action values

(‘rational agent’)

$$v^\star(s) = \max_a q^\star(s,a)$$



For the optimal policy the expectation over policy probabilities changes into a maximization

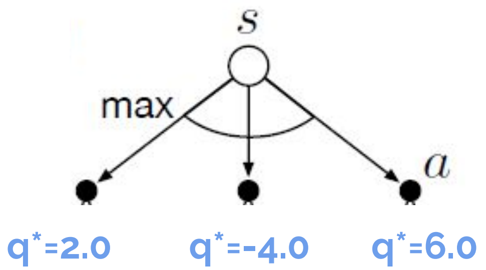
Optimal policy

<u>Key insight:</u>

The optimal policy is a greedy/max policy with respect to the optimal state-action values

('rational agent')

$$v^\star(s) = \max_a q^\star(s,a)$$



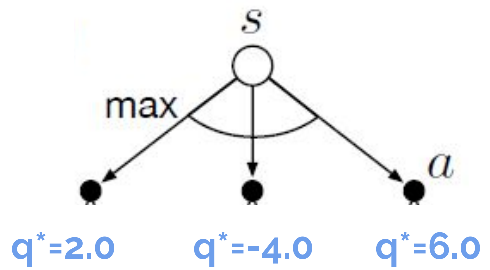
Optimal policy

<u>Key insight:</u>

The optimal policy is a greedy/max policy with respect to the optimal state-action values

('rational agent')

$$v^\star(s) = \max_a q^\star(s, a)$$



π*(a|s)= [0.0, 0.0, 1.0]

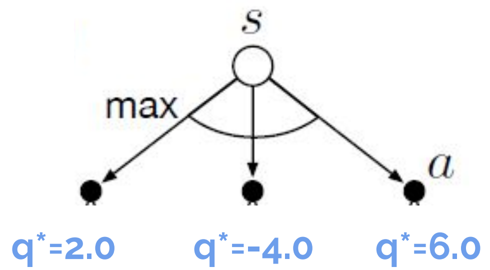
Optimal policy

<u>Key insight:</u>

The optimal policy is a greedy/max policy with respect to the optimal state-action values

('rational agent')

$$v^\star(s) = \max_a q^\star(s, a)$$



π*(a|s)= [0.0, 0.0, 1.0]

If we find the optimal values q* we also know the optimal policy
(just act greedily with respect to the values)

Bellman Optimality Equation

Bellman Optimality Equation

We can use this insight to write a specific Bellman Equation for the optimal value function

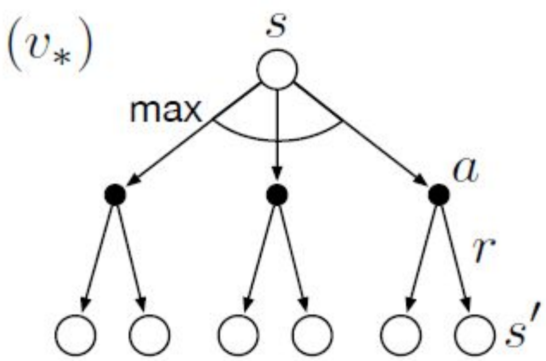
Bellman Optimality Equation for $v^*(s)$

Bellman Optimality Equation for $v^*(s)$

$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v^\star(s') \Big]$$

Bellman Optimality Equation for v*(s)

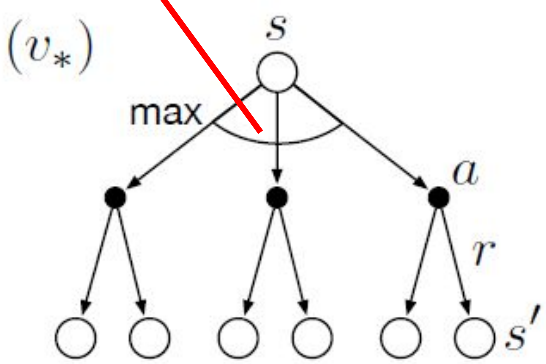
$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v^\star(s') \Big]$$



Bellman Optimality Equation for v*(s)

$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)}\Big[r(s,a,s') + \gamma \cdot v^\star(s')\Big]$$

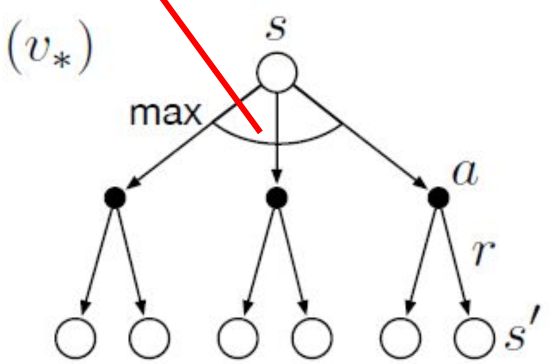
Only the max over action changed



Bellman Optimality Equation for $v^*(s)$

$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v^\star(s') \Big]$$

Only the max over action changed



This (system of) equations is <u>only</u> satisfied by the optimal state value function v*(s,a)

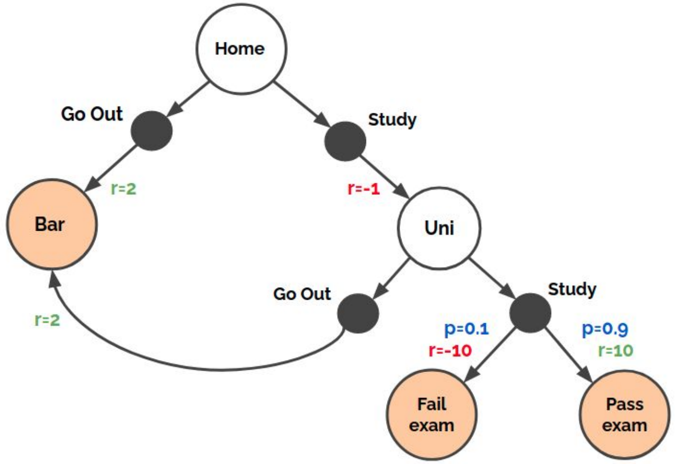
Bellman Optimality Equation: Illustration

Bellman Optimality Equation: Illustration

$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v^\star(s') \Big]$$

Bellman Optimality Equation: Illustration

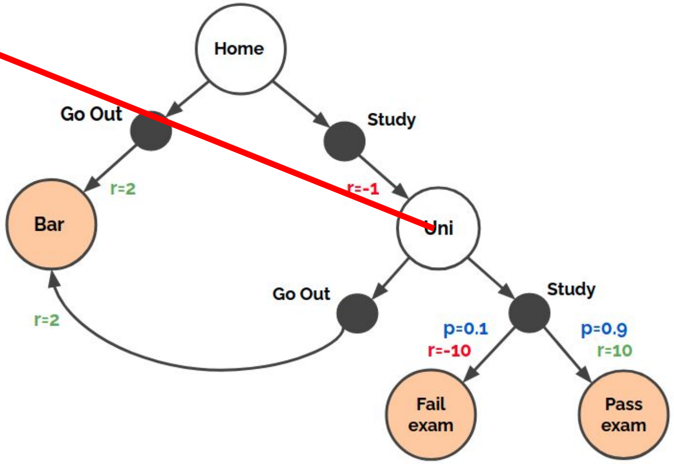
$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^\star(s') \right]$$



Bellman Optimality Equation: Illustration

$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)}\left[r(s,a,s') + \gamma \cdot v^\star(s')\right]$$

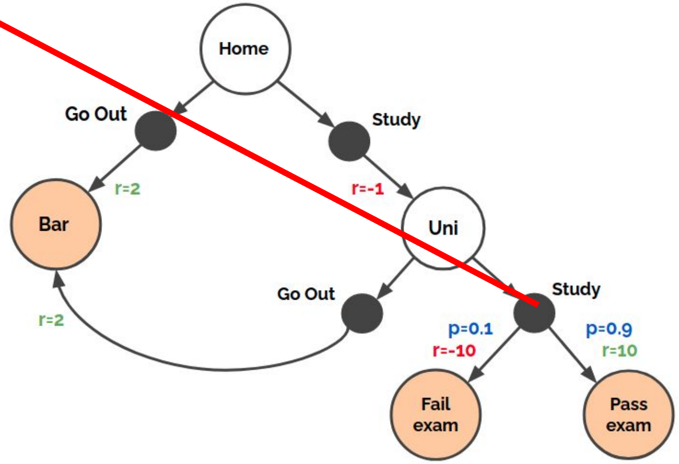
Q: How would you compute v*(Uni)?



Bellman Optimality Equation: Illustration

$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^\star(s') \right]$$

Best action is Study

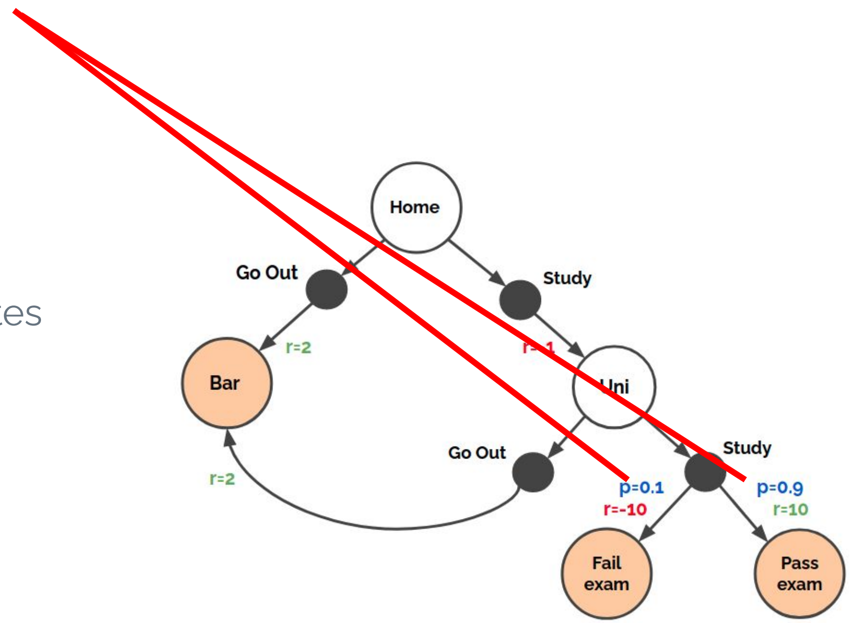


Bellman Optimality Equation: Illustration

$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^\star(s') \right]$$

Best action is Study

which may (10-90%) lead to two next states



Bellman Optimality Equation: Illustration

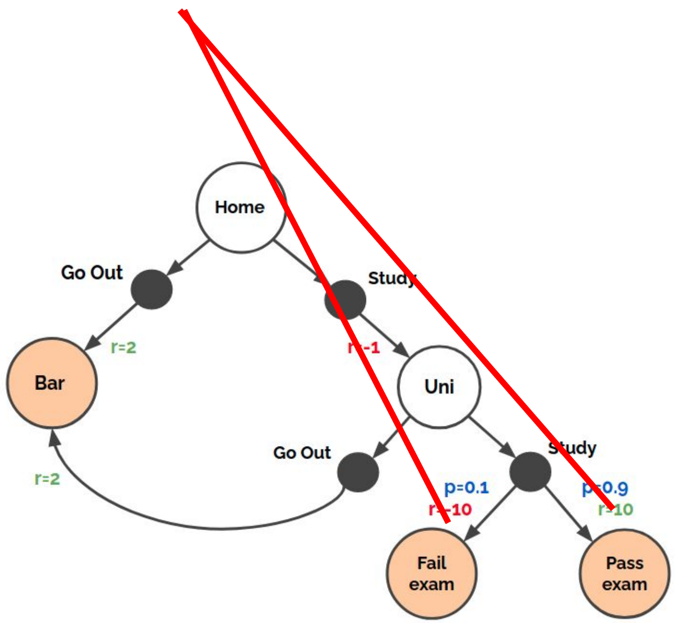
$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \left[r(s,a,s') + \gamma \cdot v^\star(s') \right]$$

Best action is Study

which may (10-90%) lead to two next states

10% gives reward of -10 plus nothing after

90% gives reward of +10 plus nothing after



Bellman Optimality Equation: Illustration

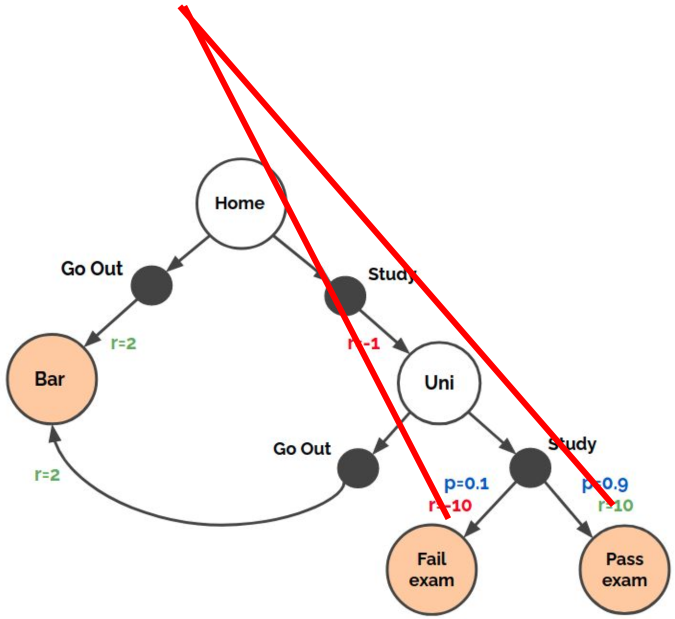
$$v^\star(s) = \max_a \mathbb{E}_{s' \sim p(s'|a,s)} \Big[r(s,a,s') + \gamma \cdot v^\star(s') \Big]$$

Best action is Study

which may (10-90%) lead to two next states

10% gives reward of -10 plus nothing after

90% gives reward of +10 plus nothing after



Bellman Optimality Equation is very intuitive:
we used it (without knowing it as an equation) at the very start of the previous lecture

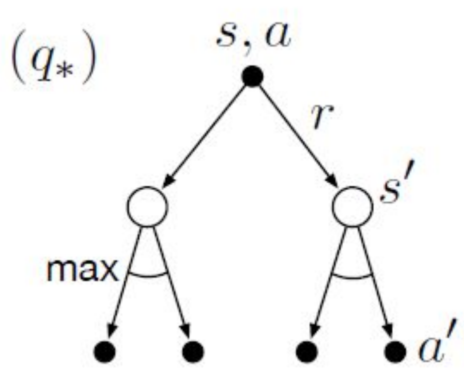
Bellman Optimality Equation for $q^*(s,a)$

Bellman Optimality Equation for $q^*(s,a)$

$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

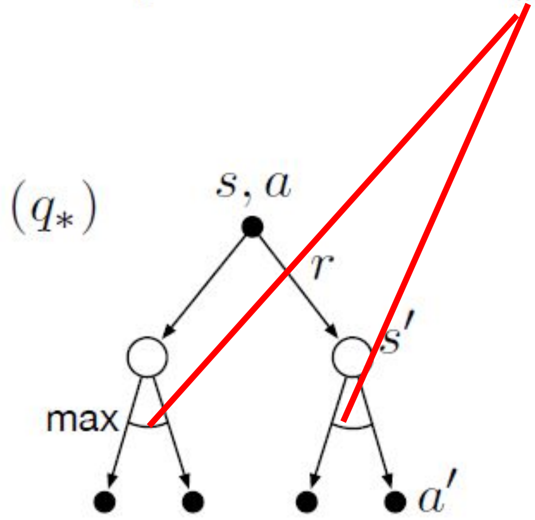
Bellman Optimality Equation for q*(s,a)

$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$



Bellman Optimality Equation for q*(s,a)

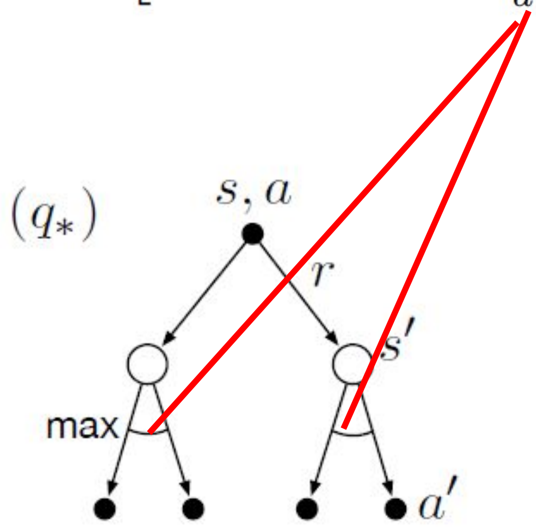
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$



Again only the max over action changed

Bellman Optimality Equation for q*(s,a)

$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$



This (system of) equations is only satisfied by the optimal state-action value function q*(s,a)

Part IVb

Value Iteration

Value Iteration

Value Iteration

If we perform Dynamic Programming (as before) but on the Bellman *Optimality* Equation,

then we converge on the optimal state(-action) value function!

Value Iteration (DP) for $q^*(s,a)$

Value Iteration (DP) for $q^*(s,a)$

Input: an MDP ($p(s'|s,a)$, $r(s,a,s')$, γ)

Value Iteration (DP) for $q^*(s,a)$

Input: an MDP (p(s'|s,a), r(s,a,s'), γ)

Algorithm:

- Initialize $q^*(s)=0$ for all s,a

Value Iteration (DP) for $q^*(s,a)$

Input: an MDP $(p(s'|s,a), r(s,a,s'), \gamma)$

Algorithm:

- Initialize $q^*(s)=0$ for all s,a
- Repeat until convergence:

Value Iteration (DP) for $q^*(s,a)$

Input: an MDP ($p(s'|s,a)$, $r(s,a,s')$, γ)

Algorithm:

- Initialize $q^*(s)=0$ for all s,a
- Repeat until convergence:
  - For each s in state space:
    - For each a in action space:

Value Iteration (DP) for q*(s,a)

Input: an MDP (p(s'|s,a), r(s,a,s'), γ)

Algorithm:

- Initialize q*(s)=0 for all s,a
- Repeat until convergence:
  - For each s in state space:
    - For each a in action space:

$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Value Iteration (DP) for $q^*(s,a)$

Input: an MDP (p(s'|s,a), r(s,a,s'), γ)

Algorithm:

- Initialize $q^*(s)=0$ for all s,a
- Repeat until convergence:
  - For each s in state space:
    - For each a in action space:

$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Very simple algorithm, but converges on the optimal value function $q^*(s,a)$

Value Iteration (DP) for $q^*(s,a)$

Input: an MDP (p(s'|s,a), r(s,a,s'), γ)

Algorithm:

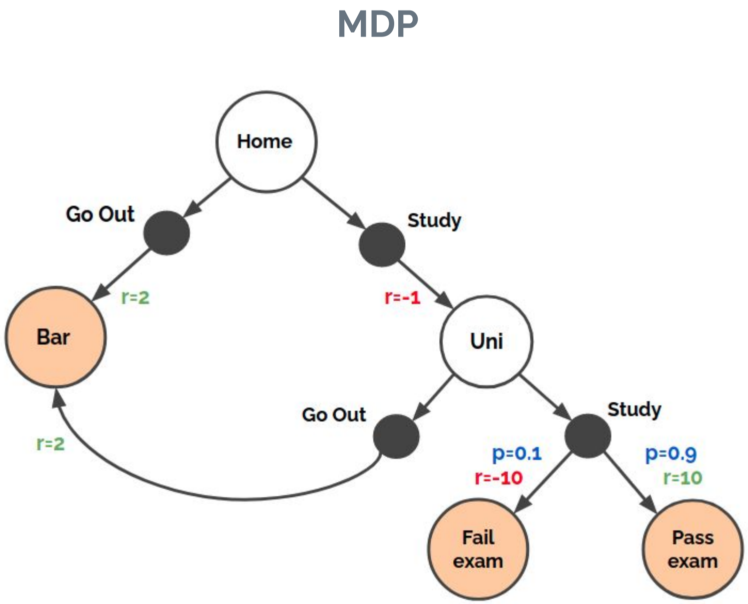
- Initialize $q^*(s)=0$ for all s,a
- Repeat until convergence:
    - For each s in state space:
        - For each a in action space:

$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Very simple algorithm, but converges on the optimal value function $q^*(s,a)$
[directly have optimal policy by acting greedy with respect to $q^*(s,a)$]

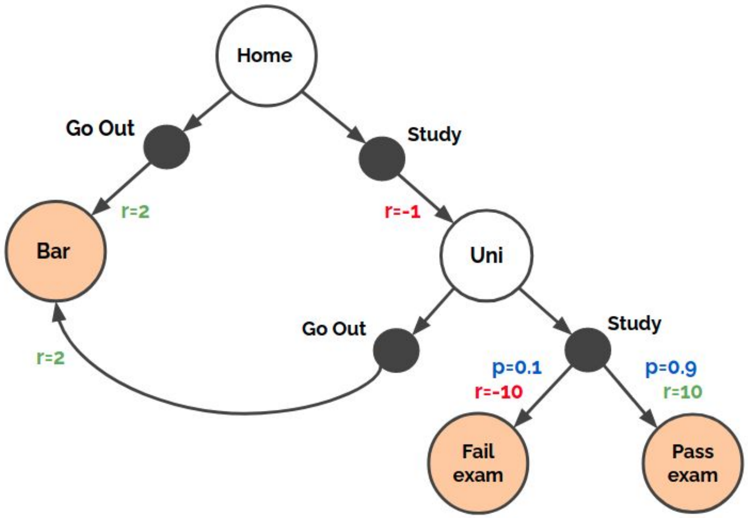
Value Iteration: Example

Value Iteration: Example



Value Iteration: Example

MDP



Bellman Optimality Equation

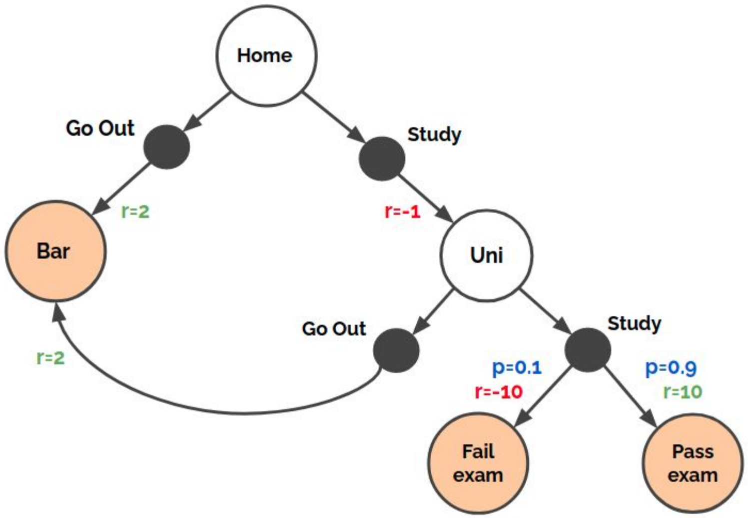
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	0.0	0.0
Bar	0.0	0.0
Uni	0.0	0.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Bellman Optimality Equation

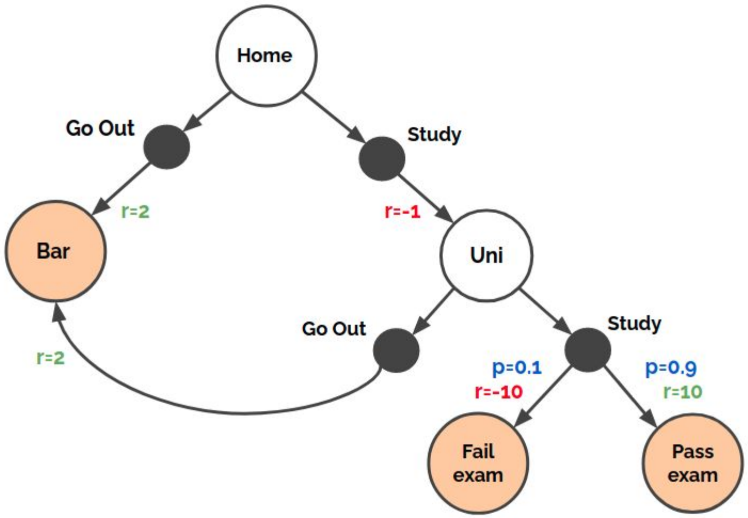
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a') \right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	-1.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



I have completed the first full sweep

Bellman Optimality Equation

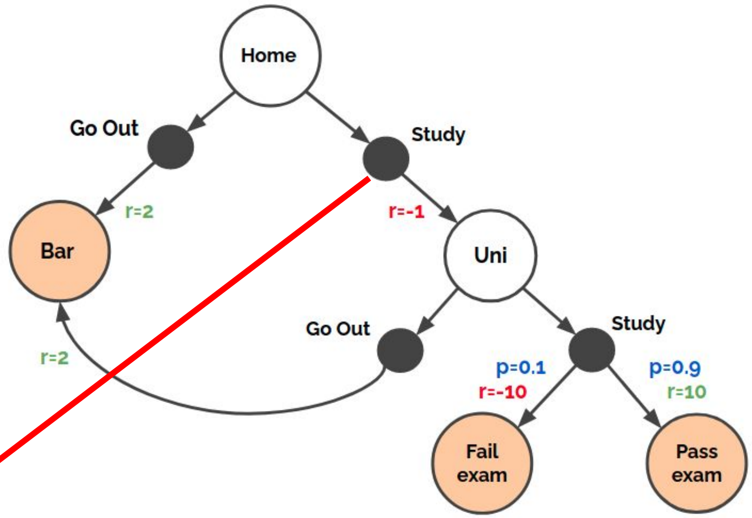
$$q^{\star}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^{\star}(s',a')\right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	-1.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Q: Update q*(Home, Study)

Bellman Optimality Equation

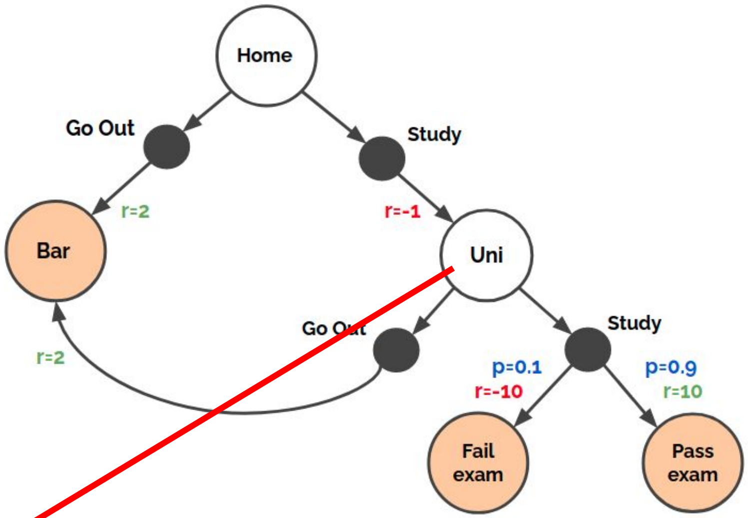
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	-1.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Q: Update q*(Home, Study)

A: Always end up at Uni

Bellman Optimality Equation

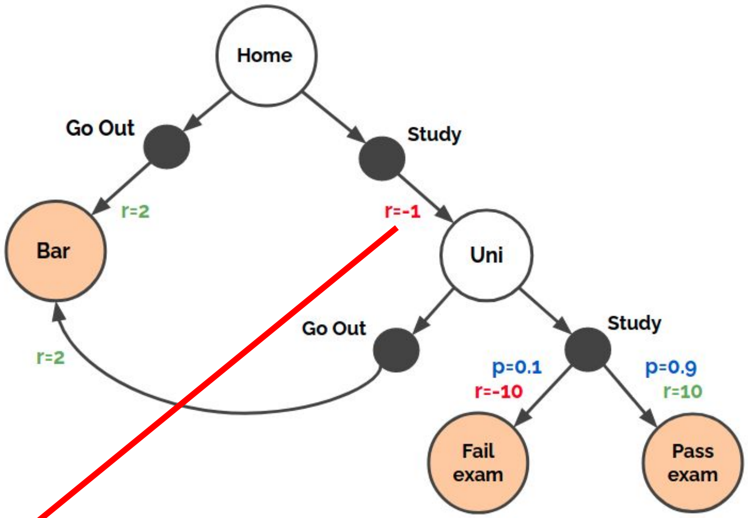
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	-1.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Q: Update q*(Home, Study)

A: Always end up at Uni
Immediate reward of -1.0

Bellman Optimality Equation

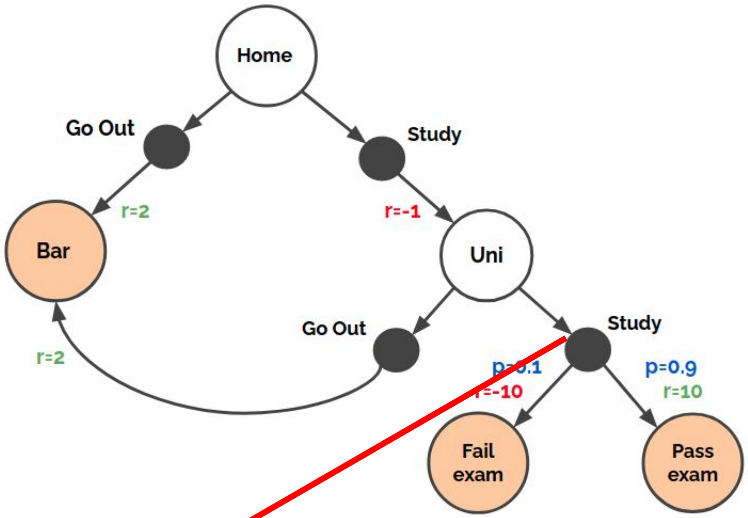
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	-1.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Q: Update q*(Home, Study)

A: Always end up at Uni
Immediate reward of -1.0
Best next action is Study

Bellman Optimality Equation

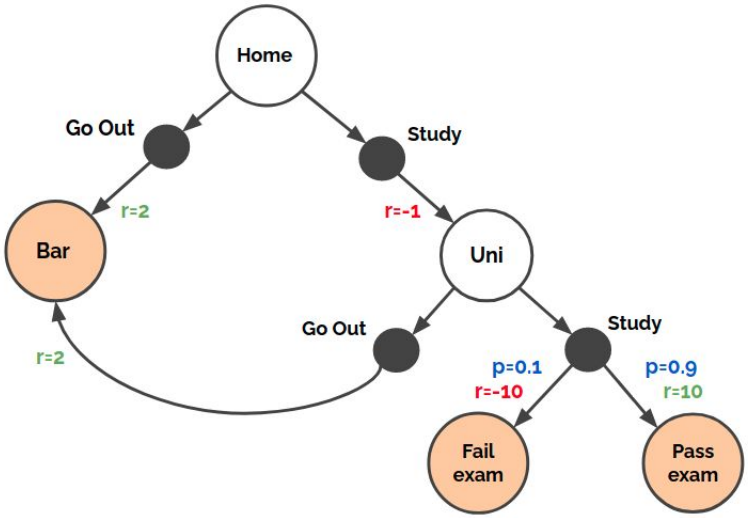
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a') \right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	-1.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Q: Update q*(Home, Study)

A: Always end up at Uni
Immediate reward of -1.0
Best next action is Study

-1.0 + 1.0 · max(2.0,8.0) = 7.0

Bellman Optimality Equation

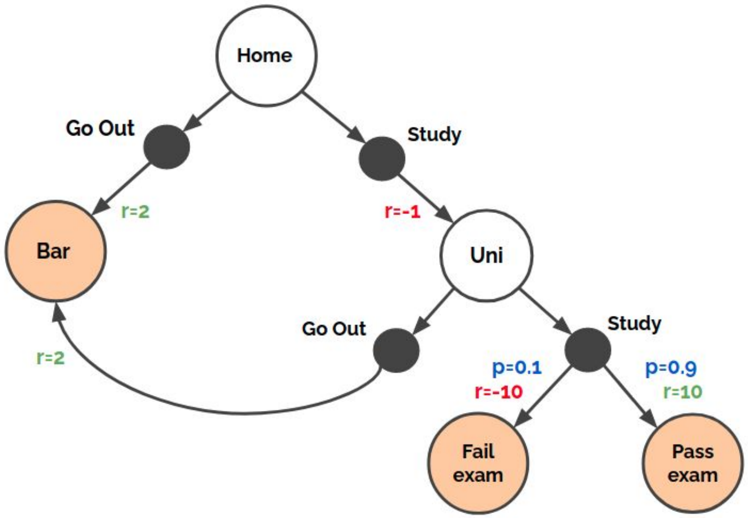
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a') \right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	7.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Q: Update q*(Home, Study)

A: Always end up at Uni
Immediate reward of -1.0
Best next action is Study

-1.0 + 1.0 · max(2.0,8.0) = 7.0

Bellman Optimality Equation

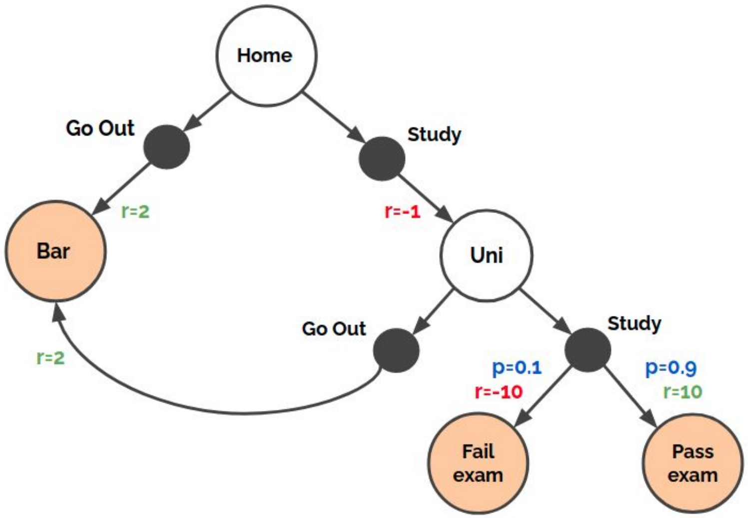
$$q^\star(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a') \right]$$

Value Iteration: Example

q*(s,a) solution table

	Go Out	Study
Home	2.0	7.0
Bar	0.0	0.0
Uni	2.0	8.0
Fail exam	0.0	0.0
Pass exam	0.0	0.0

MDP



Update next state-action, etc.

Bellman Optimality Equation

$$q^{\star}(s,a) = \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot \max_{a'} q^{\star}(s',a') \right]$$

Part IVc

Generalized Policy Iteration

Generalized Policy Iteration

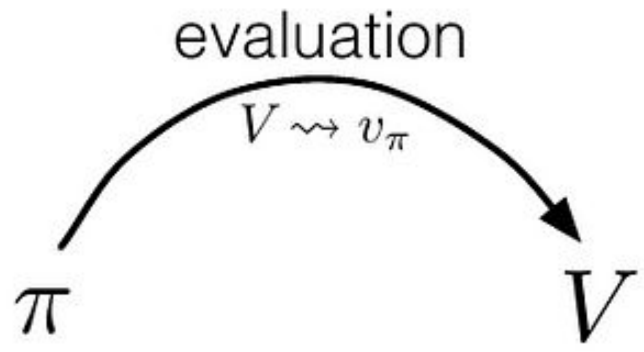
Generalized Policy Iteration

We now have all the ingredients to specify the general scheme of MDP solution algorithms

Generalized Policy Iteration

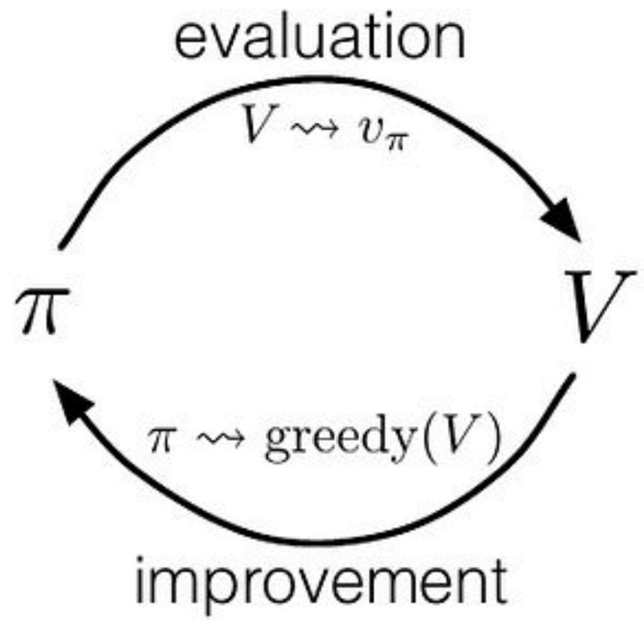
π V

Generalized Policy Iteration



Policy Evaluation: Compute the value function of a given policy

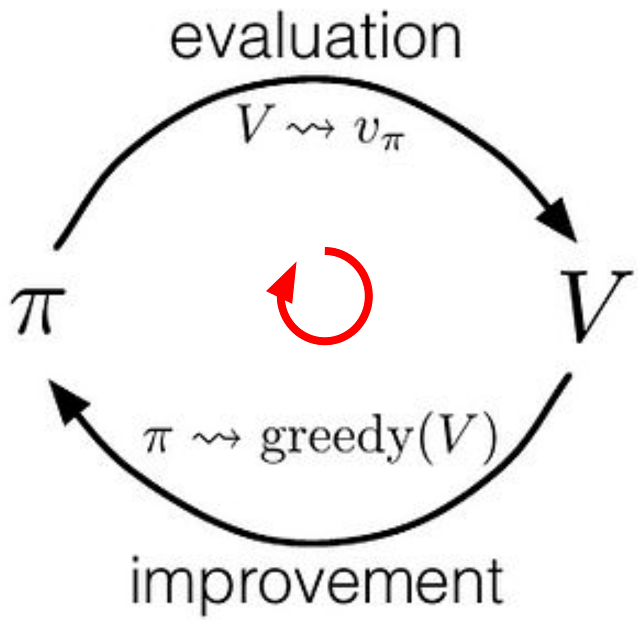
Generalized Policy Iteration



Policy Evaluation: Compute the value function of a given policy

Policy Improvement: Compute an greedy improved policy from the obtained value

Generalized Policy Iteration



Iterating these two procedures will converge on the optimal value function and policy

Policy Evaluation: Compute the value function of a given policy

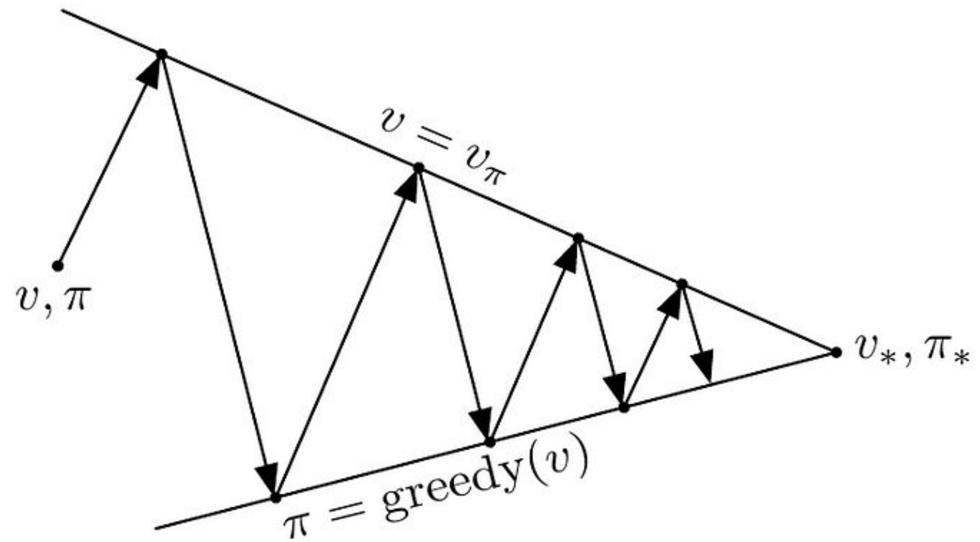
Policy Improvement: Compute an greedy improved policy from the obtained value

Generalized Policy Iteration

Iterating both procedures will converge on the optimal value function and policy

Generalized Policy Iteration

Iterating both procedures will converge on the optimal value function and policy



Part IVd

Policy Iteration

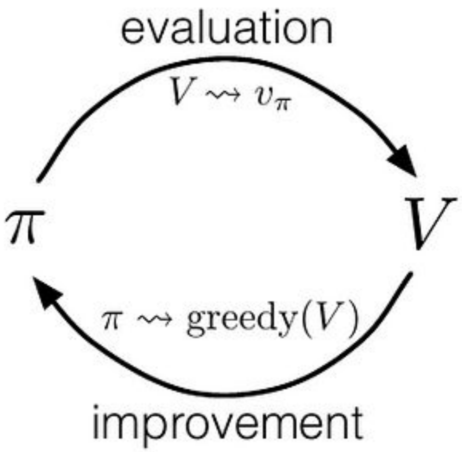
Policy Iteration (DP)

Policy Iteration (DP)

We already have all the ingredients to implement generalized policy iteration

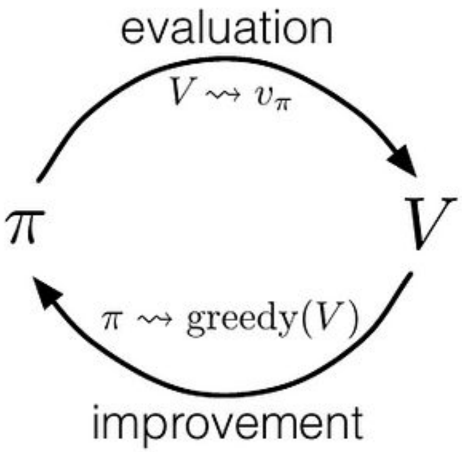
Policy Iteration (DP)

We already have all the ingredients to implement generalized policy iteration



Policy Iteration (DP)

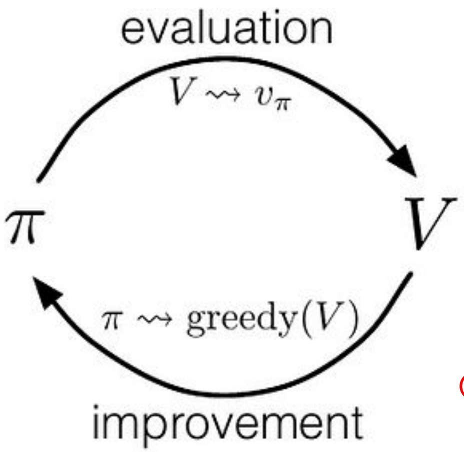
We already have all the ingredients to implement generalized policy iteration



Dynamic Programming on Bellman Equation (Sec. II)

Policy Iteration (DP)

We already have all the ingredients to implement generalized policy iteration

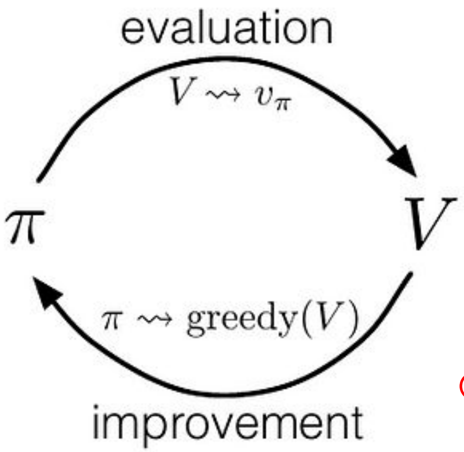


Dynamic Programming on Bellman Equation (Sec. II)

Greedy policy improvement (Sec. III)

Policy Iteration (DP)

We already have all the ingredients to implement generalized policy iteration



Dynamic Programming on Bellman Equation (Sec. II)

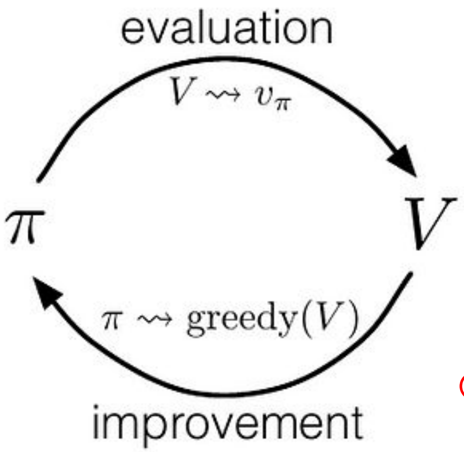
Greedy policy improvement (Sec. III)

'Policy Iteration"

Converges on the optimal value function and policy

Special trick

What if we only do one sweep of policy evaluation (instead of until convergence)

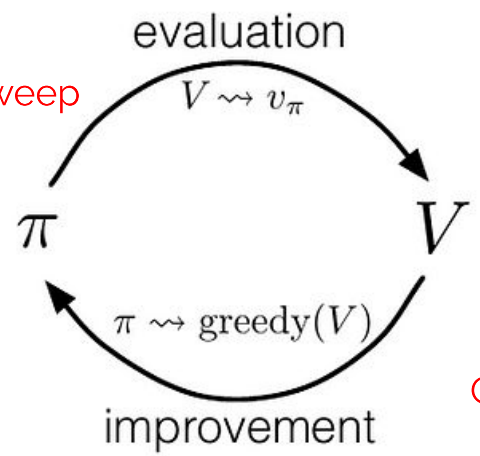


Dynamic Programming on
Bellman Equation (Sec. II)

Greedy policy improvement (Sec. III)

Special trick

What if we only do one sweep of policy evaluation (instead of until convergence)



Special trick

Repeat until convergence:

1. Policy evaluation (one sweep)

$$q^{\pi}(s,a) \leftarrow \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \mathbb{E}_{a' \sim \pi(a'|s')}[q^{\pi}(s',a')]\right]$$

2. Policy improvement

$$\pi(s) \leftarrow \arg\max_{a} q^{\pi}(s,a)$$

Special trick

Repeat until convergence:

1. Policy evaluation (one sweep)

$$q^{\pi}(s,a) \leftarrow \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot \mathbb{E}_{a' \sim \pi(a'|s')} [q^{\pi}(s',a')] \right]$$

2. Policy improvement

$$\pi(s) \leftarrow \arg\max_{a} q^{\pi}(s,a)$$

Can write both updates in a single equation

(verify this at home)

Special trick

Repeat until convergence:

$$q^\star(s,a) \leftarrow \mathbb{E}_{s' \sim p(s'|s,a)}\Big[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\Big]$$

Special trick

Repeat until convergence:

$$q^\star(s,a) \leftarrow \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

= Bellman Optimality Equation

Value Iteration (DP)

Repeat until convergence:

$$q^\star(s,a) \leftarrow \mathbb{E}_{s' \sim p(s'|s,a)}\left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a')\right]$$

Value Iteration (DP)

Repeat until convergence:

$$q^\star(s,a) \leftarrow \mathbb{E}_{s' \sim p(s'|s,a)} \left[r(s,a,s') + \gamma \cdot \max_{a'} q^\star(s',a') \right]$$

Value iteration (Dynamic Programming on the Bellman Optimality Equation, Sec. IVb)

is a

special case of generalized policy iteration with a single sweep of policy evaluation

Summary

Summary

Can implement generalized policy iteration with dynamic programming in two ways:

Summary

Can implement generalized policy iteration with dynamic programming in two ways:

1. **Policy Iteration**:
   a. Policy evaluation on Bellman Equation (until convergence)
   b. Greedy policy improvement

Summary

Can implement generalized policy iteration with dynamic programming in two ways:

1. **Policy Iteration**:
   a. Policy evaluation on Bellman Equation (until convergence)
   b. Greedy policy improvement

2. **Value iteration**:
   a. Policy evaluation on Bellman Equation (single sweep)
   b. Greedy policy improvement

Summary

Can implement generalized policy iteration with dynamic programming in two ways:

1. **Policy Iteration**:
   a. Policy evaluation on Bellman Equation (until convergence)
   b. Greedy policy improvement

2. **Value iteration**:
   a. Policy evaluation on Bellman Equation (single sweep)
   b. Greedy policy improvement

Reduces to DP on Bellman Optimality Equation

Next Block

Next Block

- Dynamic Programming requires a (descriptive) model of the MDP: $p(s'|s,a)$, $r(s,a,s')$
  - In most real-world tasks these are hard to obtain

Next Block

- Dynamic Programming requires a (descriptive) model of the MDP: $p(s'|s,a)$, $r(s,a,s')$
    - In most real-world tasks these are hard to obtain
- Instead we do often have a simulator: an environment in which we can sample traces from some start state
    - The real-world also falls in this category

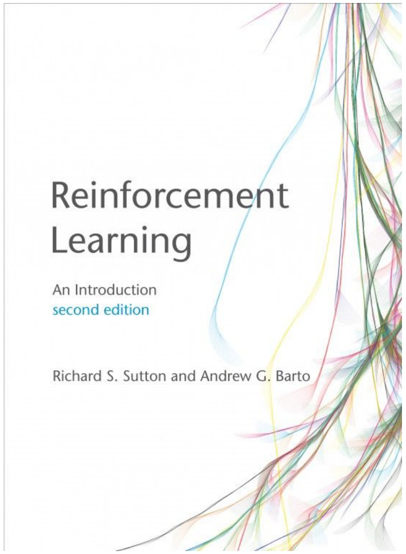
Next Block

- Dynamic Programming requires a (descriptive) model of the MDP: p(s'|s,a), r(s,a,s')
    - In most real-world tasks these are hard to obtain
- Instead we do often have a simulator: an environment in which we can sample traces from some start state
    - The real-world also falls in this category
- We can still learn good policies and value functions from sampled traces
    - Known as **reinforcement learning**

At Home (read)

At Home (read)

- Sutton & Barto Chapter 4
- Lecture slides & notes

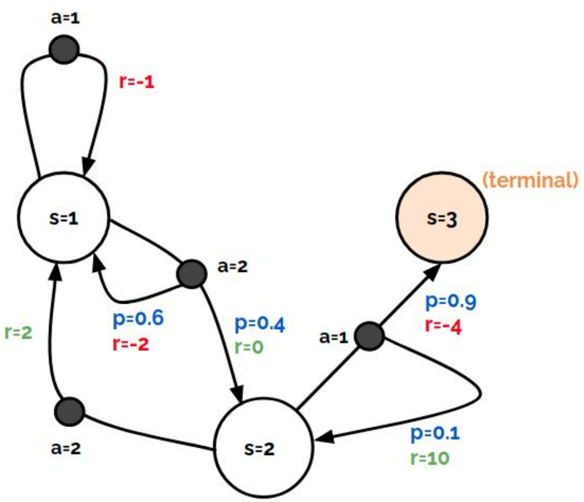


http://incompleteideas.net/book/RLbook2020.pdf

At Home (by hand)

At Home (by hand)

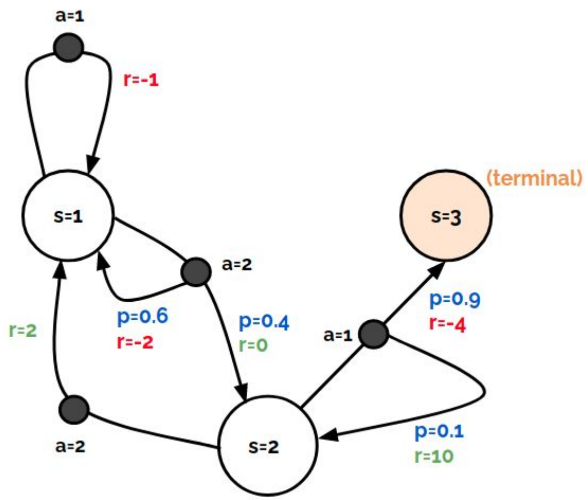
1. Draw your own MDP without loops



At Home (by hand)

1. Draw your own MDP without loops

2. Compute the optimal policy for your MDP
   a. Work backwards from terminal states
   b. Update with max over actions, expectation over dynamics (Bellman Optimality Equation)

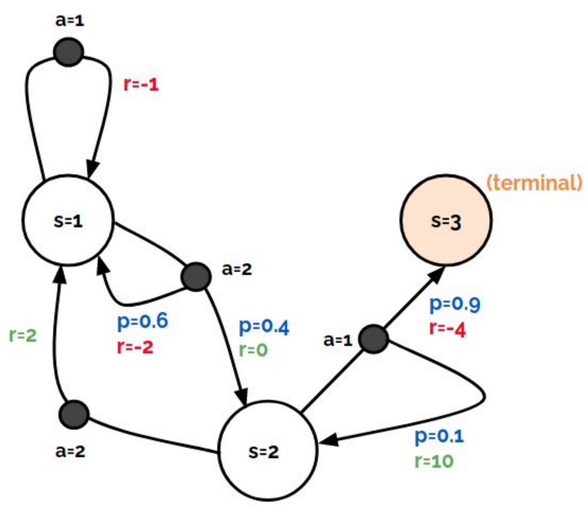


At Home (by hand)

1. Draw your own MDP without loops

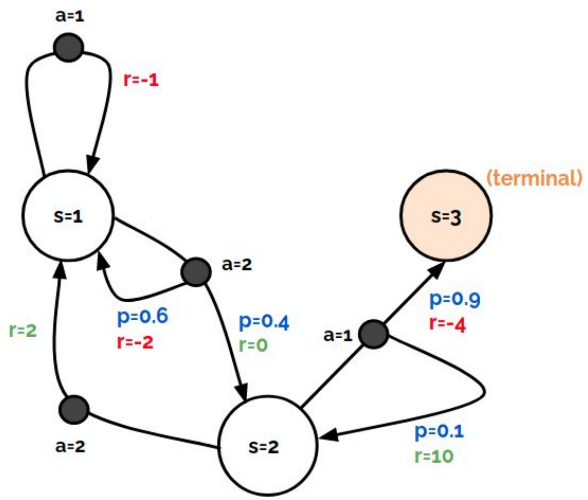
2. Compute the optimal policy for your MDP
   a. Work backwards from terminal states
   b. Update with max over actions, expectation over dynamics (Bellman Optimality Equation)

3. Extend your MDP to include loops
   a. Optimal value function no longer trivial to compute



At Home (by hand)

1. Draw your own MDP without loops
2. Compute the optimal policy for your MDP
   a. Work backwards from terminal states
   b. Update with max over actions, expectation over dynamics (Bellman Optimality Equation)
3. Extend your MDP to include loops
   a. Optimal value function no longer trivial to compute
4. Make a rough guess for the state(-action) value function



At Home (code)

At Home (code)

Go to Colab: http://tiny.cc/ntbjvz

At Home (code)

Go to Colab: http://tiny.cc/ntbjvz

Work through the notebook examples of dynamic programming (policy evaluation, value iteration, policy iteration)

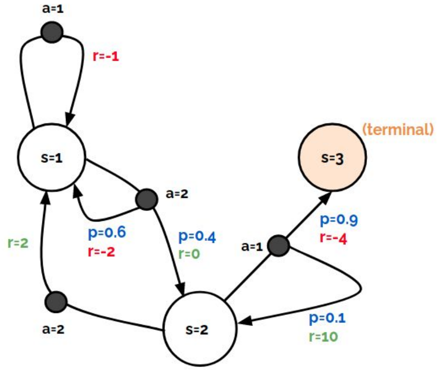
At Home (code)

Go to Colab: http://tiny.cc/ntbjvz

Work through the notebook examples of dynamic programming (policy evaluation, value iteration, policy iteration)

Implement your designed MDP with loops:

- Run value/policy iteration on it.
- How many iterations do you need till convergence?
- How close was your guessed value function?



Questions?