Reinforcement Learning III

Final Report

- Header
- Abstract
- Introduction
- Background
- Data
- Methods
- Results
- Conclusions
- Roles
- Timeline of Activity
- References
- 2,500 words max (excluding references, roles, timeline of activity)
- Complete grading rubric will be posted this week

Other Final Project Deliverables

- Video
  - Content is key – make sure you tell a clear story
  - Is it engaging and make use of the visual medium? Great opportunity to use images
  - Up to 5 min (NO LONGER)
- Github Repo
  - Contains a descriptive README.md file that explains what the repo is for, and how to use the code to reproduce your work (including how to set it up to run)
  - Is well commented throughout all files
  - Lists all dependencies in a requirements.txt file
  - Informs the user how to get the data and includes all preprocessing code
  - It actually runs and does what it says
- Peer Evaluation (criteria on the website

Do…

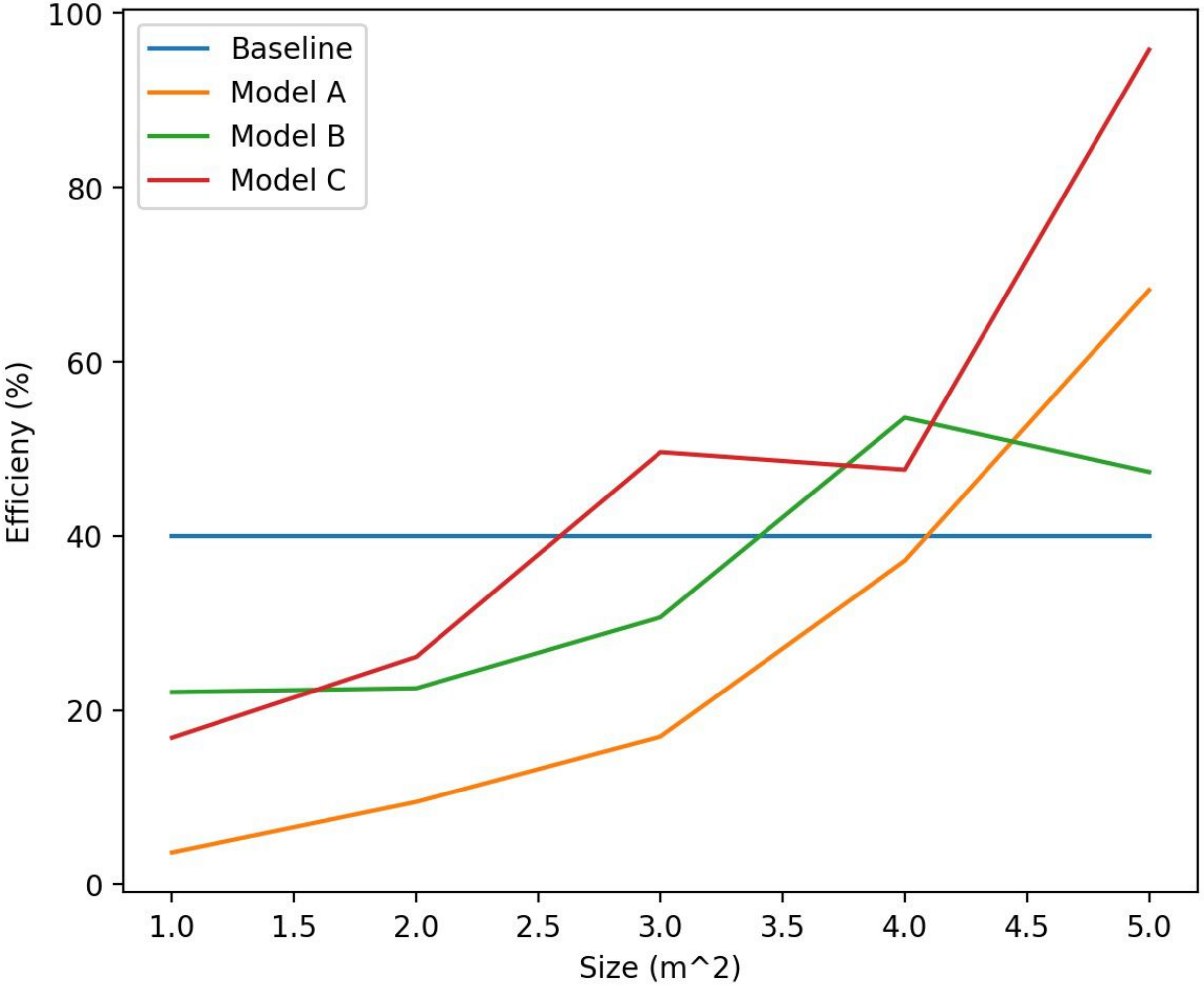
1. If you can show it as a figure, **do it**
2. Tell a story, don't write a logbook
3. Avoid unnecessary equations
4. Make your results/findings/conclusions clear and relate them back to your motivating problem/question. Avoid overly vague conclusions.
5. Make sure every figure has a contribution
6. Use flowcharts / diagrams to explain processes
7. Write professionally: avoid colloquialisms, check for spelling, grammar, punctuation, etc. Also check figure caption text
8. Reference your figures in the paper (e.g. Figure 1….). If a figure is not referenced, it doesn't have a point
9. Be precise in your language. For example: what do you mean when you say "better" – better how and in comparison to what?

Don't…

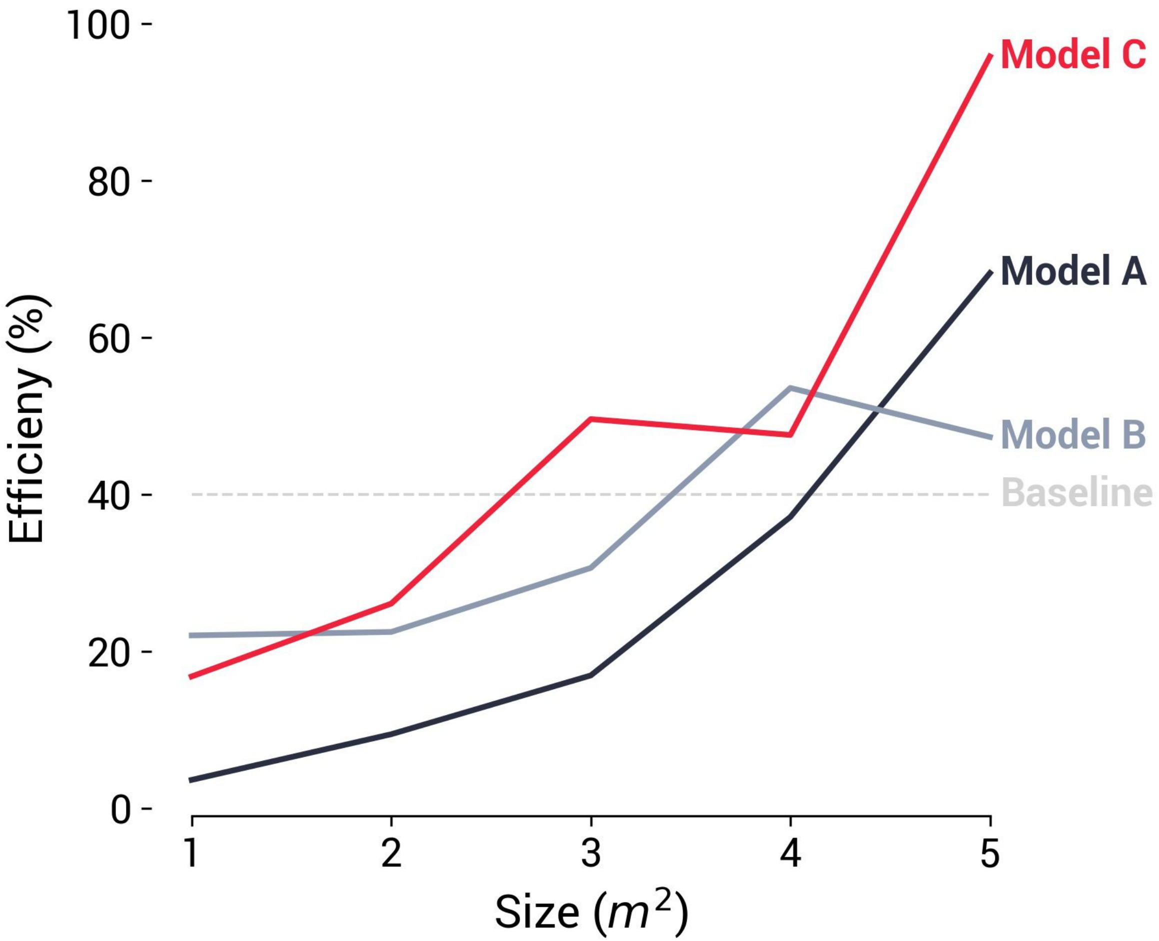
1. Do **not** use screenshots, code snippets, or variable names from code in the report or video. Use figures or tables for code output
2. Do not use random pictures of ML model architectures unless it makes a point in your report

Make your content EASY to understand

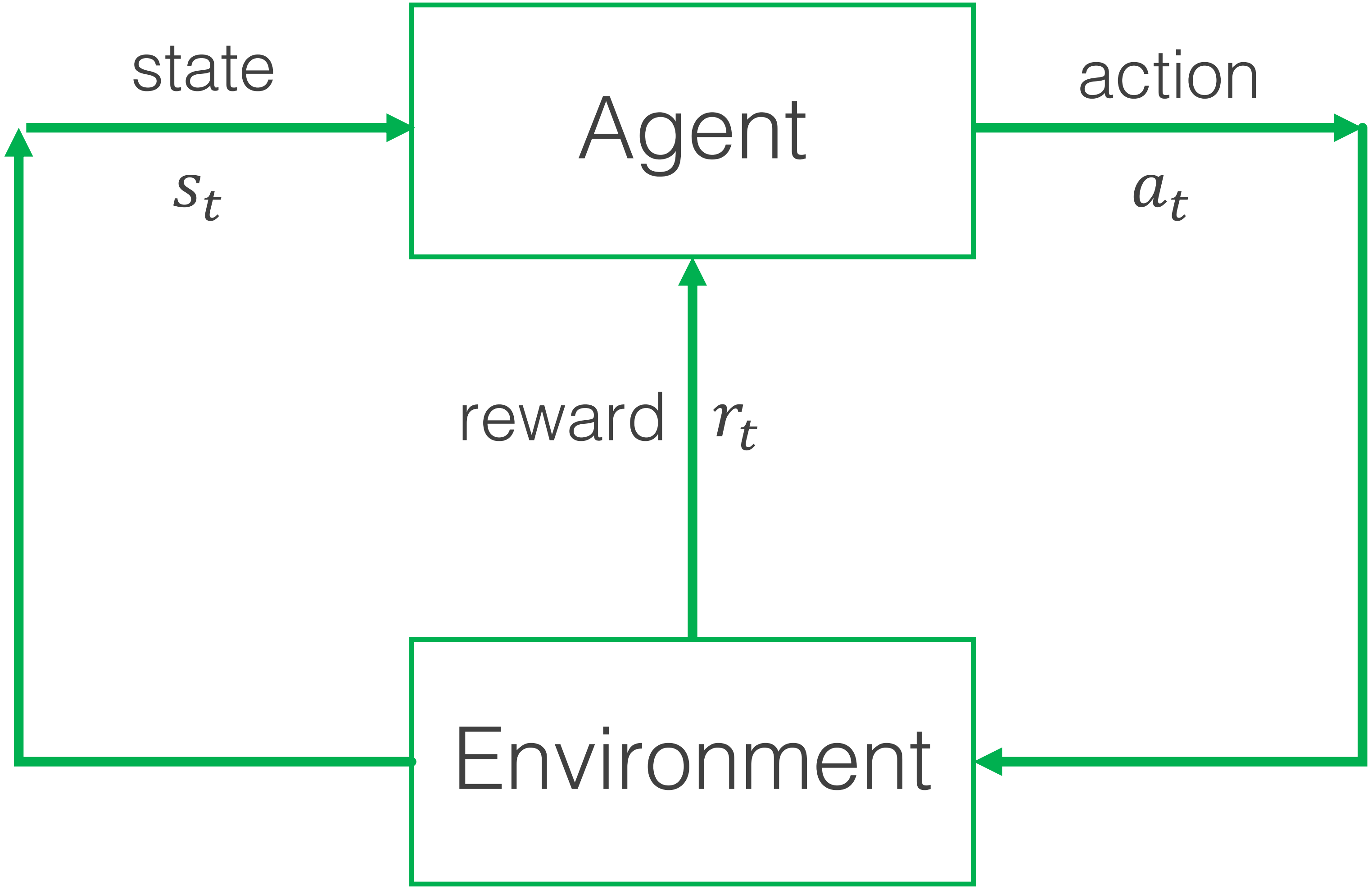
Satisfactory



Exceptional



Agent-environment Interaction



Agent at each step t…

Encounters state, s_t
Executes action a_t
Receives scalar reward, r_{t+1}

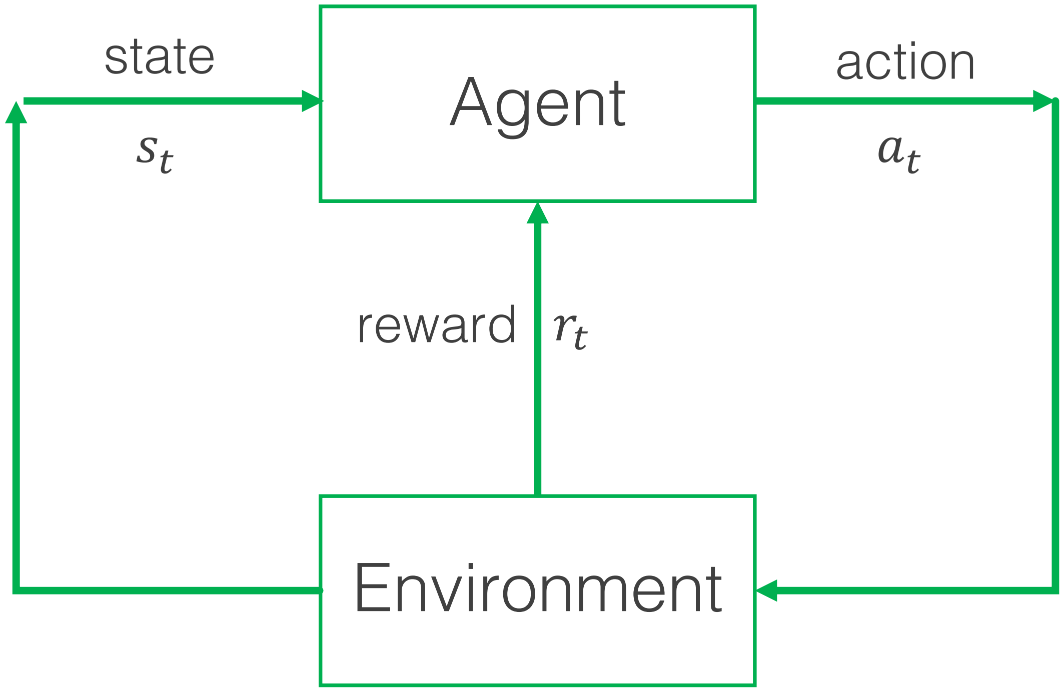
Environment at each step t…

Receives action a_t
Transitions to state, s_{t+1}
Emits scalar reward, r_{t+1}

Actions: choices made by the agent
States: basis on which choices are made
Rewards: define the agent's goals

David Silver, 2015

Reinforcement Learning Components



Policy (agent behavior), $\pi(s)$

Reward function (the goal), r_t

Value functions (expected returns),
$v(s)$ State value
$q(s,a)$ Action value

Markov Models

	States are **Fully Observable**	States are **Partially Observable**
Autonomous (no actions; make predictions)	Markov Chain, Markov Reward Process	Hidden Markov Model (HMM)
Controlled (can take actions)	Markov Decision Process (MDP)	Partially Observable Markov Decision Process (POMDP)

Applications

HMMs: time series ML, e.g. speech + handwriting recognition, bioinformatics

MDPs: used extensively for reinforcement learning

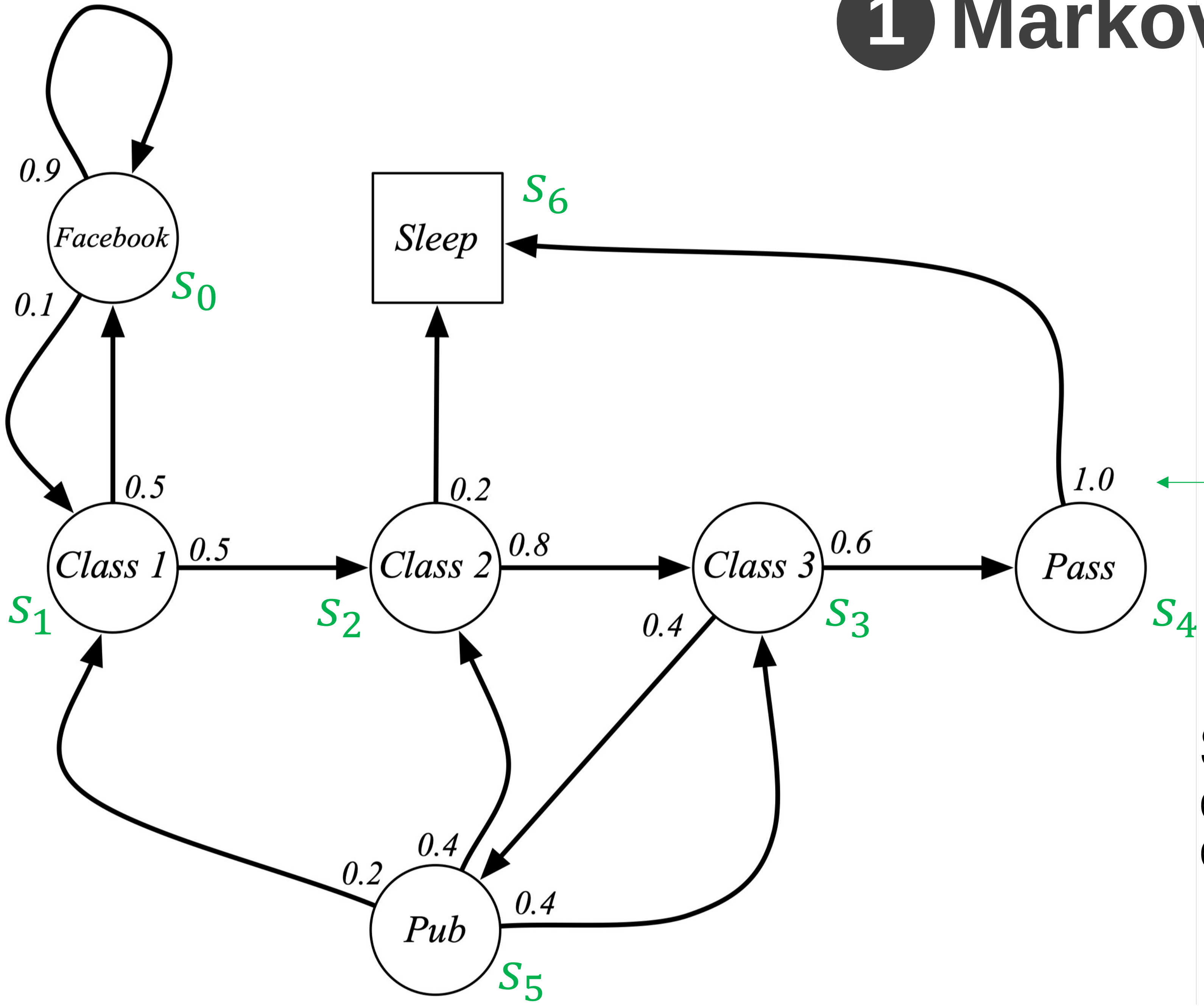
Building blocks for the full RL problem

1	Markov Chain	{state space S, transition probabilities P}
2	Markov Reward Process (MRP)	$\{S, P,$ + rewards R, discount rate $\gamma\}$ adds rewards (and values)
3	Markov Decision Process (MDP)	$\{S, P, R, \gamma,$ + actions $A\}$ adds decisions (i.e. the ability to control)

MDPs form the basis for most reinforcement learning environments

Adapted from David Silver, 2015

1 Markov Chain Example



Components:
State space S,
Transition probabilities P

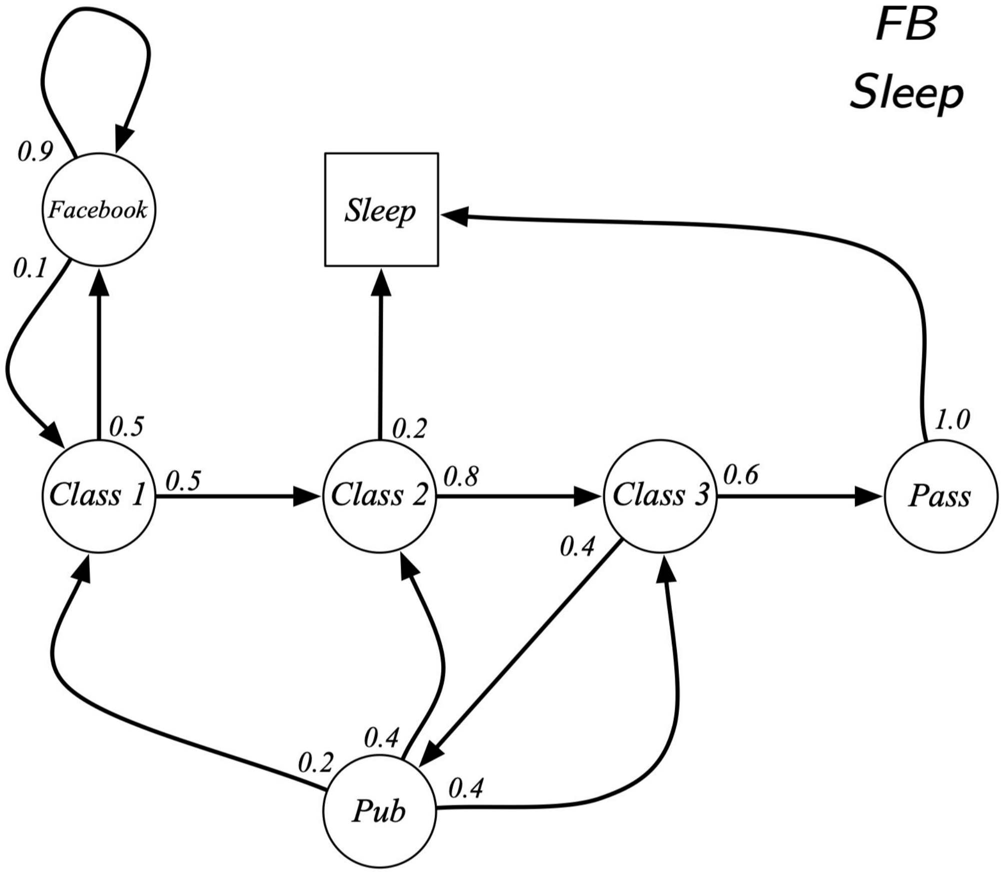
$P_{46} = P_{ss'}$

Sample Episodes:
C1,C2,Sleep
C1,FB,FB,FB,C1,C2,C3,Pass,Sleep

Example from David Silver, UCL, 2015

Markov Chain

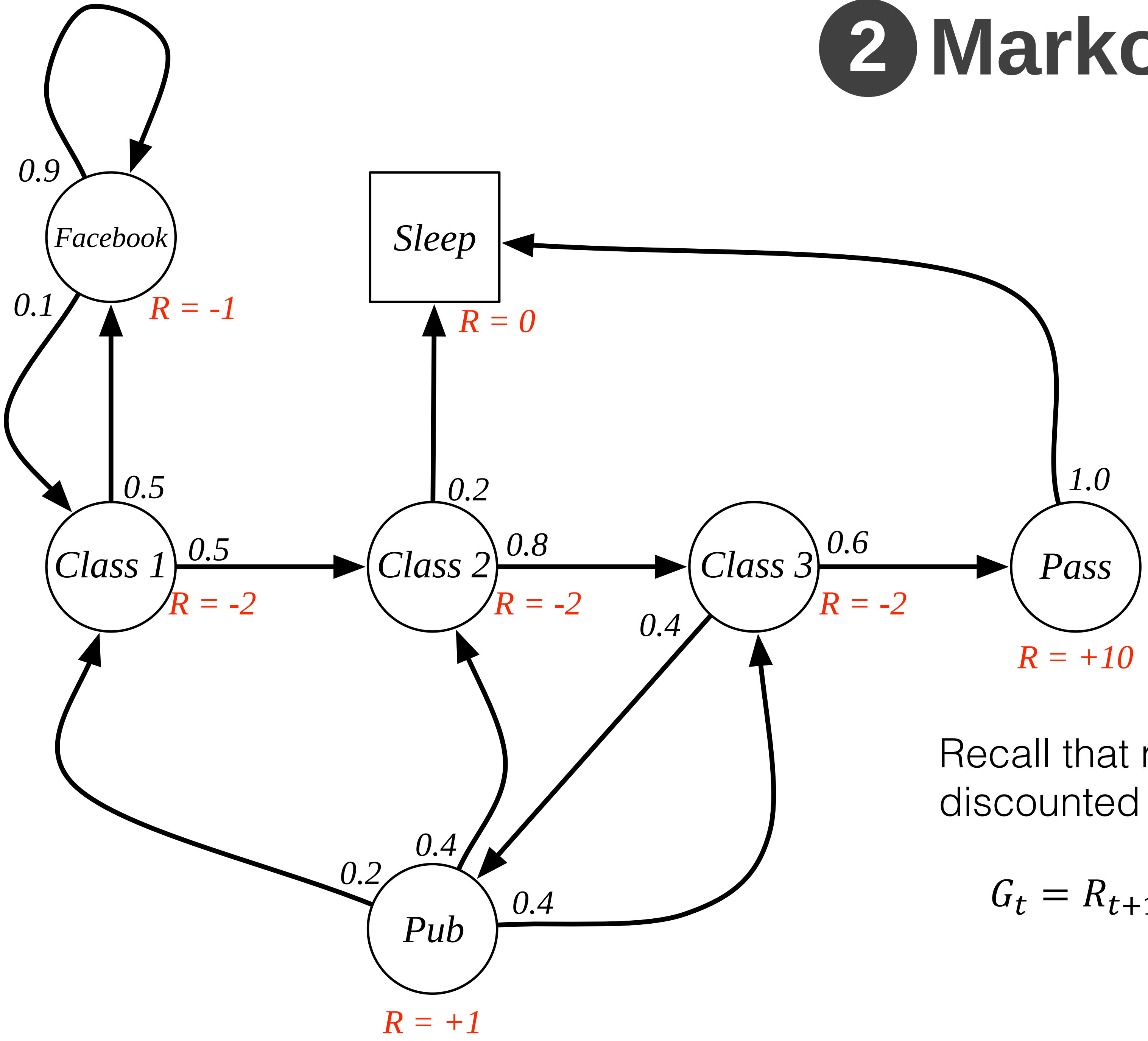
$$\mathcal{P} = \begin{array}{c} \\ C1 \\ C2 \\ C3 \\ Pass \\ Pub \\ FB \\ Sleep \end{array} \begin{array}{c} \begin{array}{ccccccc} C1 & C2 & C3 & Pass & Pub & FB & Sleep \end{array} \\ \left[\begin{array}{ccccccc} & 0.5 & & & & 0.5 & \\ & & 0.8 & & & & 0.2 \\ & & & 0.6 & 0.4 & & \\ & & & & & & 1.0 \\ 0.2 & 0.4 & 0.4 & & & & \\ 0.1 & & & & & 0.9 & \\ & & & & & & 1 \end{array}\right] \end{array}$$



State transition probability matrix, $P_{ss'}$

Example from David Silver, UCL, 2015

2 Markov Reward Process



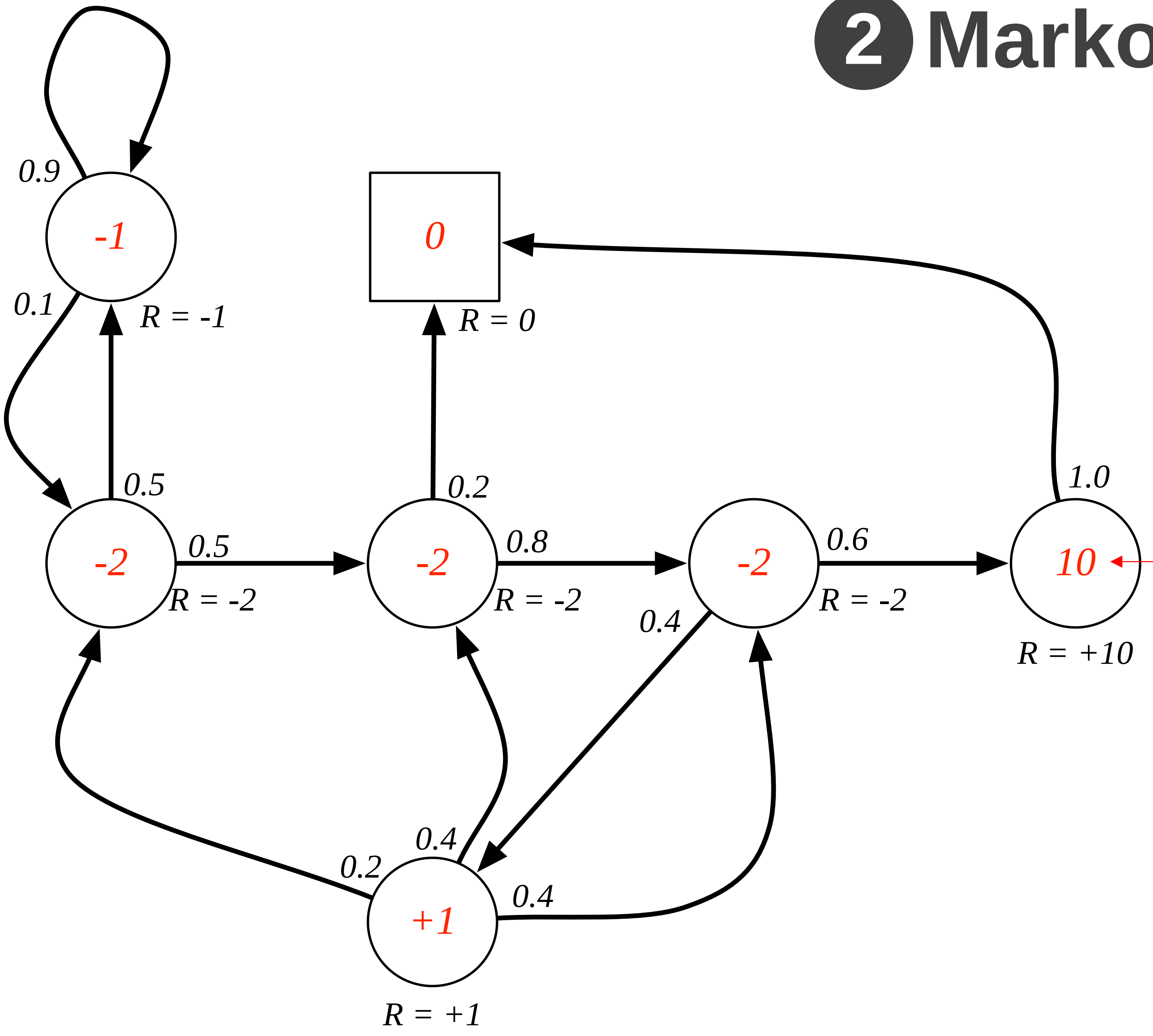
Components:
State space S,
Transition probabilities, P
Rewards, R
Discount rate, γ

Recall that returns, let's call G_t, are the total discounted rewards from time t:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Example from David Silver, UCL, 2015

❷ Markov Reward Process



Components:
State space S,
Transition probabilities, P
Rewards, R
Discount rate, γ

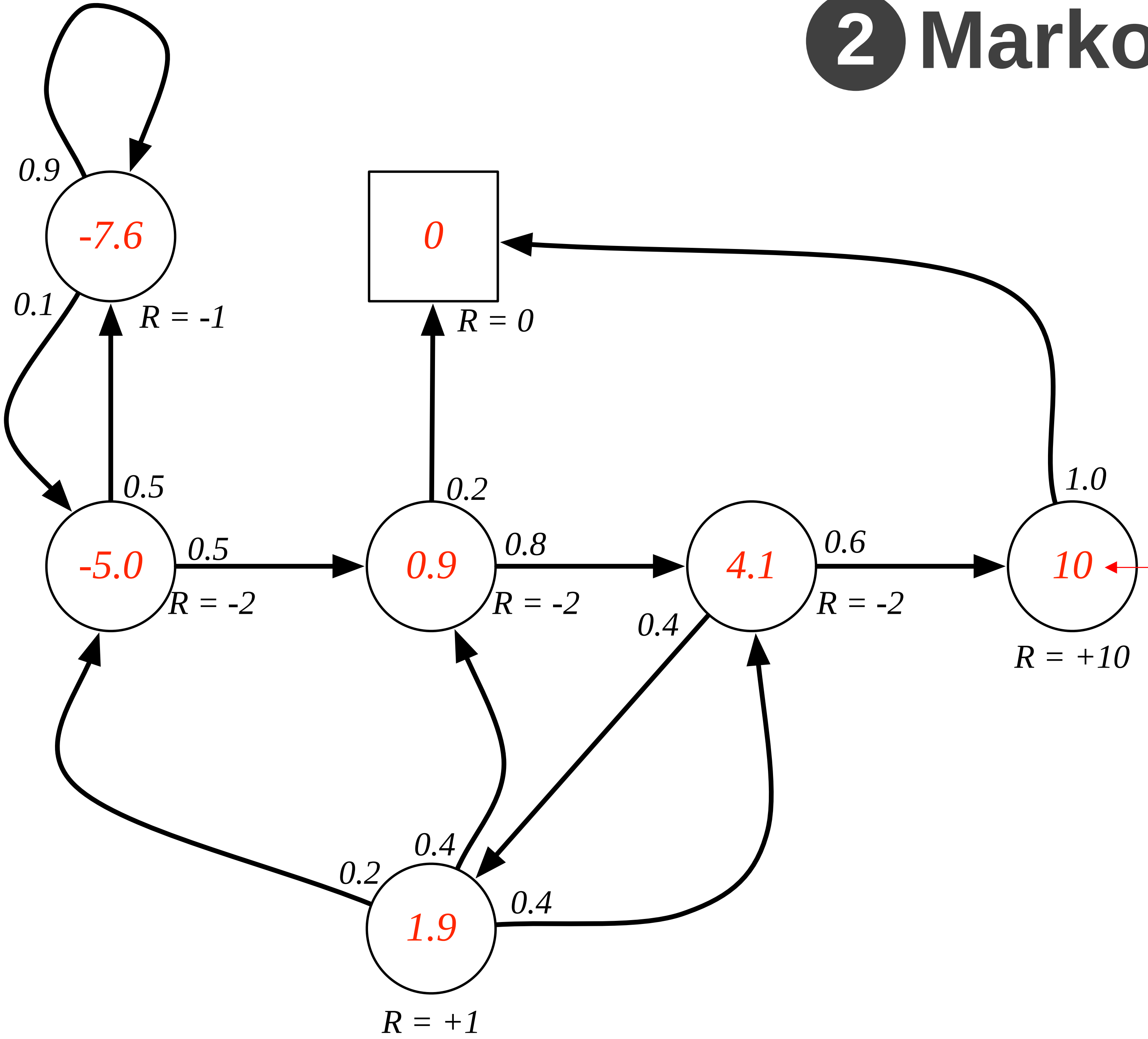
$v(s)$ for $\gamma = 0$

State value function $v(s)$ is the expected total reward (into the future)

$$v(s) = E[G_t | S = s_t]$$

Example from David Silver, UCL, 2015

❷ Markov Reward Process



Components:
State space S,
Transition probabilities, P
Rewards, R
Discount rate, γ

$v(s)$ for $\gamma = 0.9$

State value function $v(s)$ is the expected total reward (into the future)

$$v(s) = E[G_t | S = s_t]$$

Example from David Silver, UCL, 2015

"Backup" property of state value functions

$$v(s_t) = E[G_t | S = s_t] \quad \text{where } G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \dots$$

$$= E[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \dots | S = s_t]$$

$$= E[R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} \dots) | S = s_t]$$

$$= E[R_{t+1} + \gamma G_{t+1} | S = s_t]$$

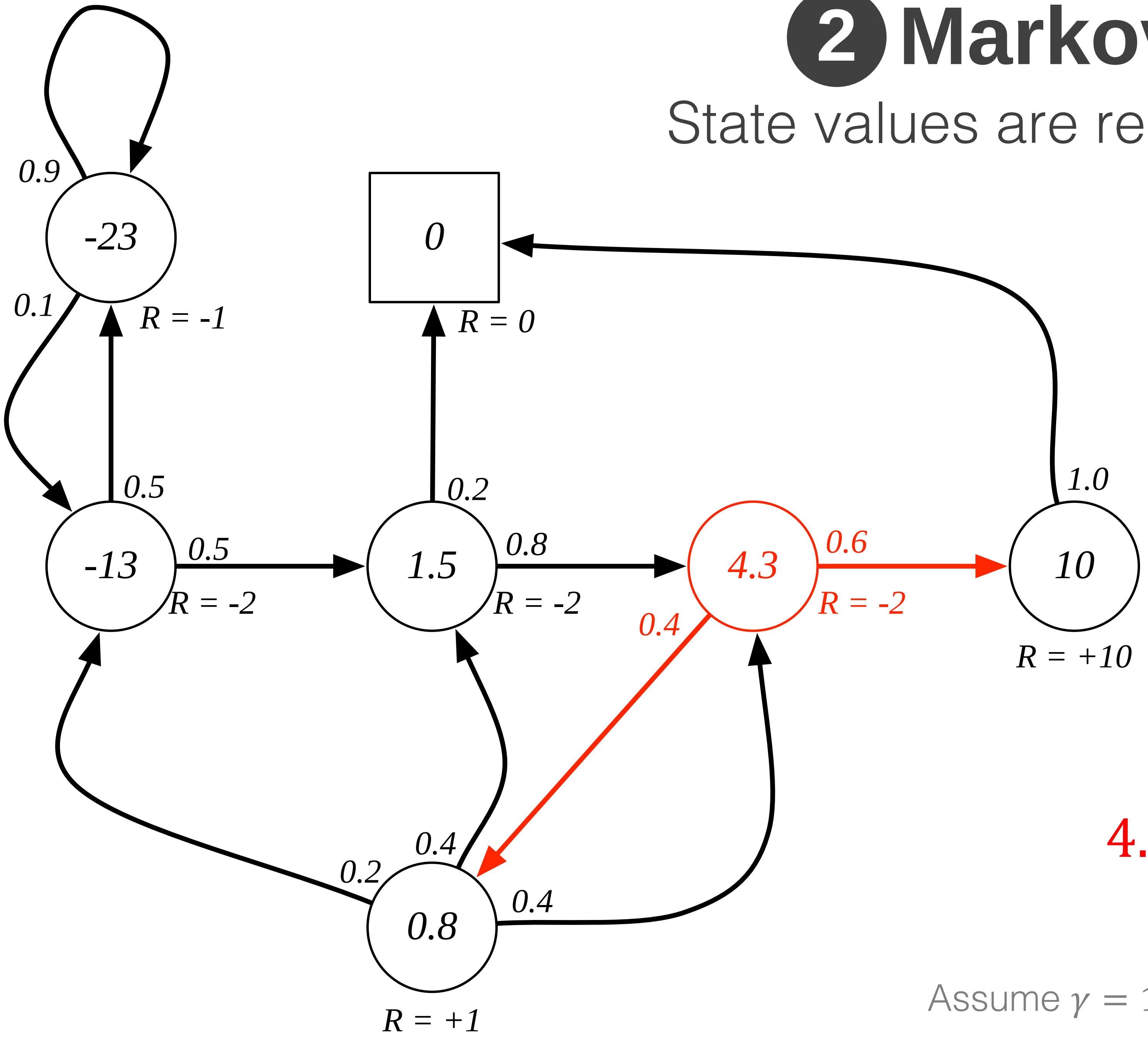
$$= E[R_{t+1} + \gamma v(s_{t+1}) | S = s_t]$$

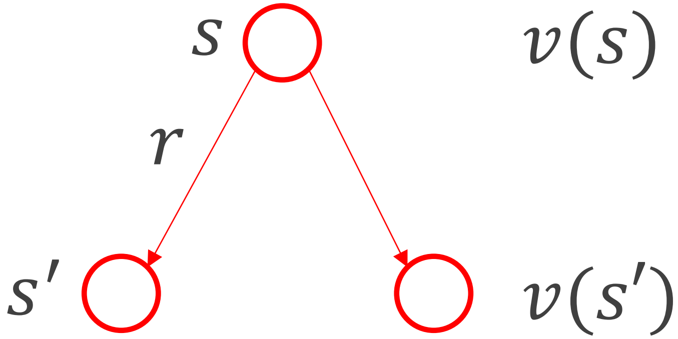
This recursive relationship is a version of the **Bellman Equation**

Example from David Silver, UCL, 2015

2 Markov Reward Process

State values are related to neighboring states





possible states we could transition to from s

$$v(s) = E[R_s + \gamma v(s')|s]$$

$$v(s) = R_s + \gamma \sum_{s'} P_{ss'} v(s')$$

$$4.3 = -2 + 0.6 \times 10 + 0.4 \times 0.8$$

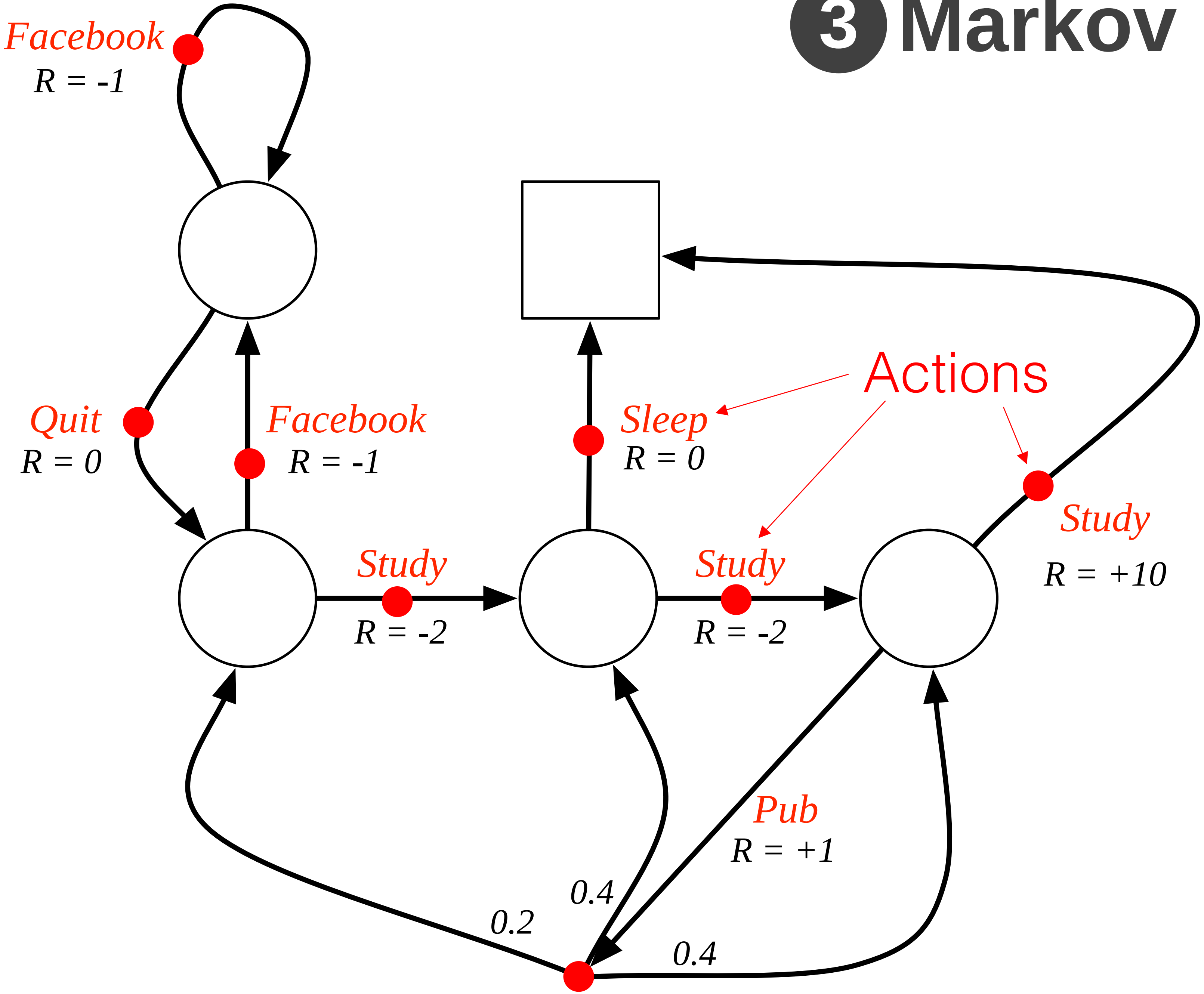
Assume $\gamma = 1$

Notation: $s = s_t$ and $s' = s_{t+1}$

$R_s = E[R_{t+1}|S_t = s]$

Example from David Silver, UCL, 2015

3 Markov Decision Process



Components:

State space S,

Transition probabilities, P

Rewards, R

Discount rate, γ

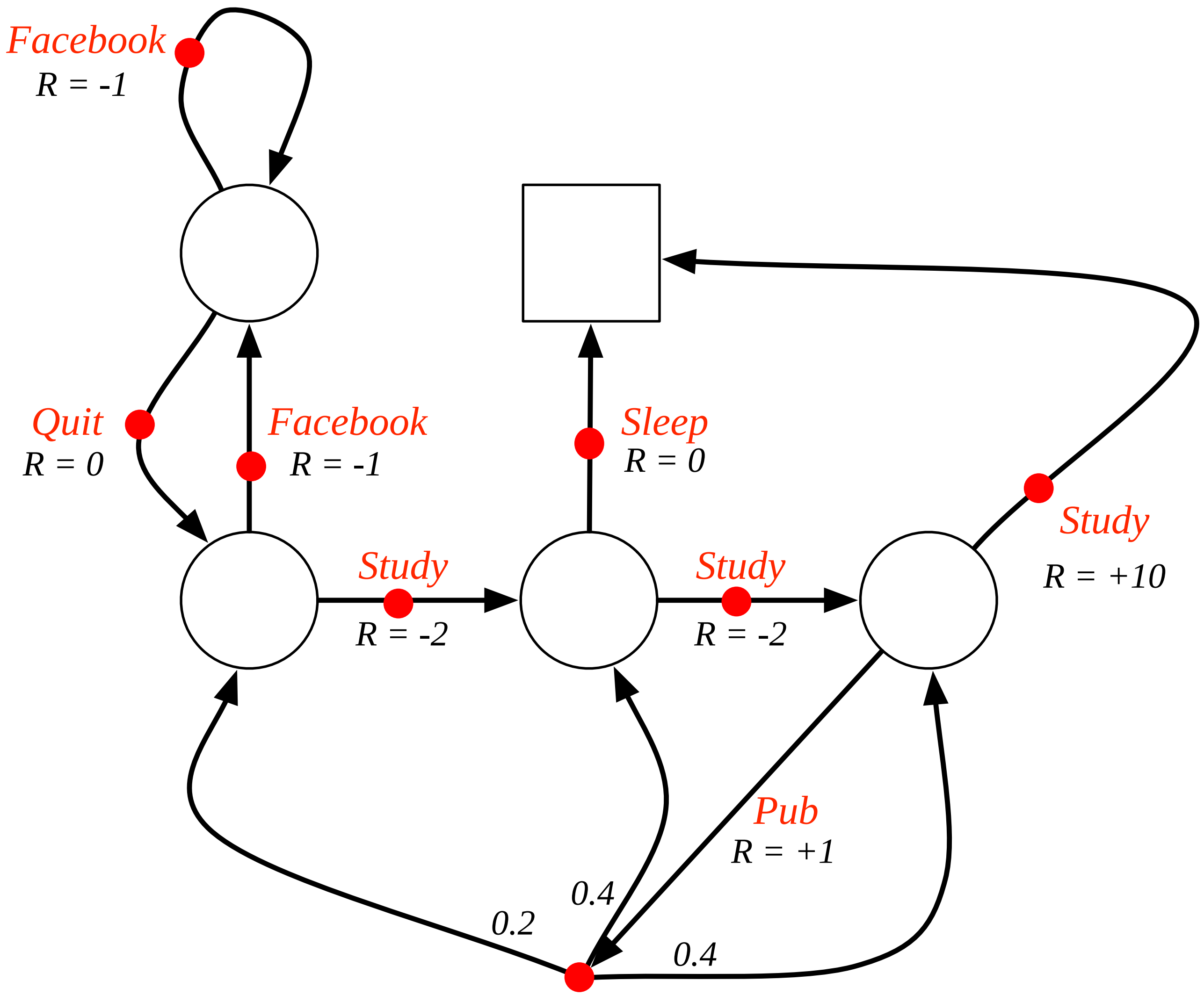
Actions, A

Adds interaction with the environment

An agent in a state chooses an action, the environment (the MDP) provides a reward and the next state

Example from David Silver, UCL, 2015

3 Markov Decision Process



Policy (how we choose actions)

(can be stochastic or deterministic)

$$\pi(a|s) = P(a|s)$$

State value function

(expected return from state s, and following policy π)

$$v_\pi(s) = E[G_t|s]$$

$$v_\pi(s) = E[R_s^a + \gamma v_\pi(s')|s]$$

Action value function

(expected return from state s, taking action a, and following policy π)

$$q_\pi(s,a) = E[G_t|s,a]$$

$$q_\pi(s,a) = E[R_s^a + \gamma q_\pi(s',a')|s,a]$$

$R_s^a = E[r_{t+1}|S_t = s, A_t = a]$

Example from David Silver, UCL, 2015

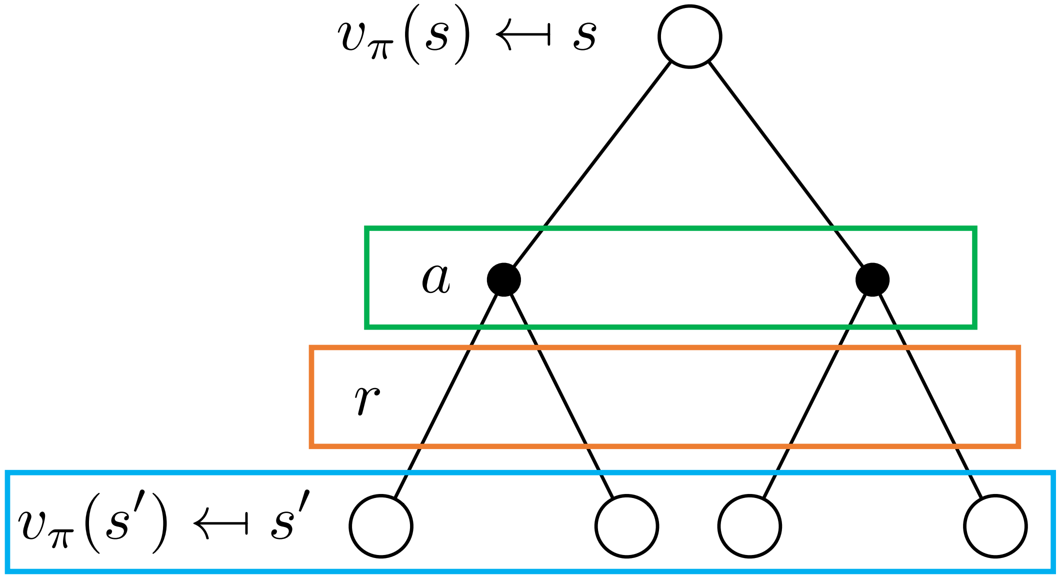
Bellman Expectation Equations for the state value function

(expected return from state s, and following policy π)

$$v_\pi(s) = E[G_t|s]$$

$$v_\pi(s) = E[R_s^a + \gamma v_\pi(s')|s]$$

$$R_s^a = E[R_{t+1}|S_t = s, A_t = a]$$



Expectation over the possible actions

Expectation over the rewards

(based on state and choice of action)

Expectation over the next possible states

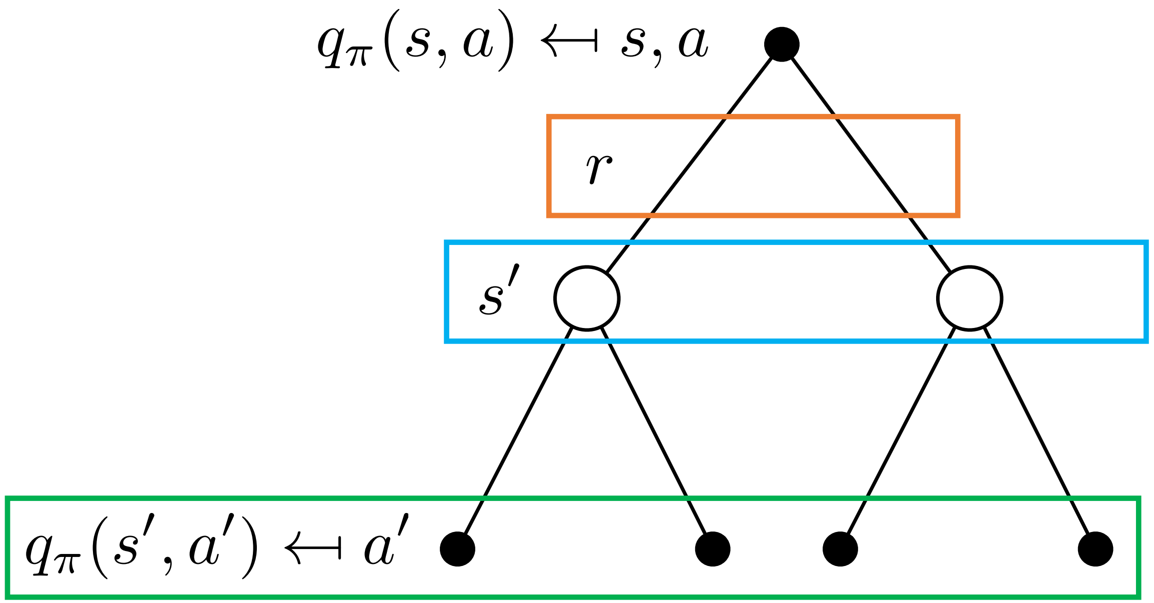
$$v_\pi(s) = \underbrace{\sum_a \pi(a|s)}\left(\underbrace{R_s^a} + \gamma \underbrace{\sum_{s'} P_{ss'}^a v_\pi(s')}\right)$$

Bellman Expectation Equations for the action value function

(expected return from state s, taking action a, then following policy π)

$$q_\pi(s,a) = E[G_t | s, a]$$
$$q_\pi(s,a) = E[R_s^a + \gamma q_\pi(s', a') | s, a]$$

$$R_s^a = E[R_{t+1} | S_t = s, A_t = a]$$



Expectation over the rewards

(based on state and choice of action)

Expectation over the next possible states

Expectation over the possible actions

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a \sum_{a'} \pi(a'|s') q_\pi(s', a')$$

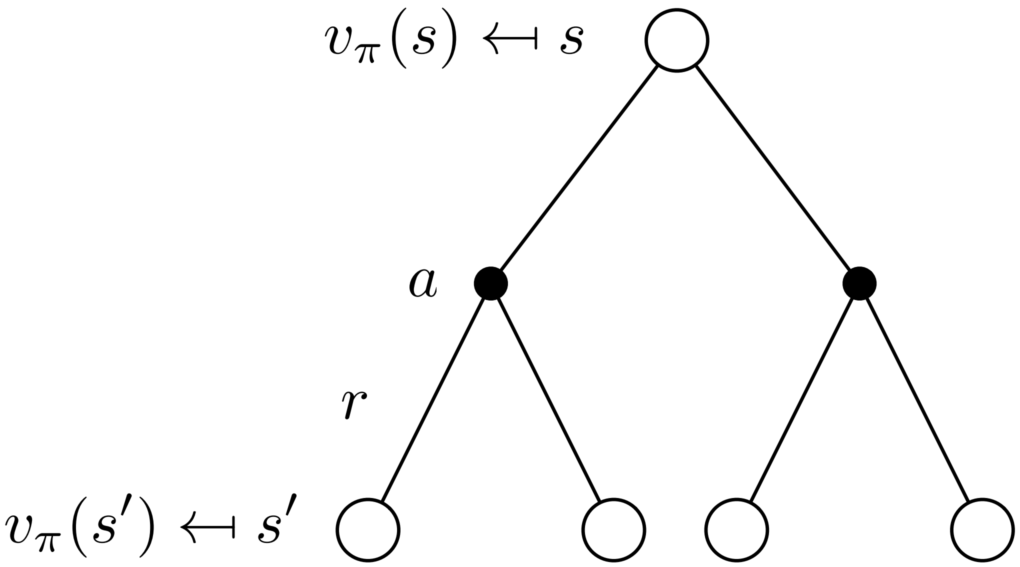
Bellman Expectation Equations

State value function

(expected return from state s, and following policy π)

$$v_\pi(s) = E[G_t|s]$$

$$v_\pi(s) = E[R_s^a + \gamma v_\pi(s')|s]$$



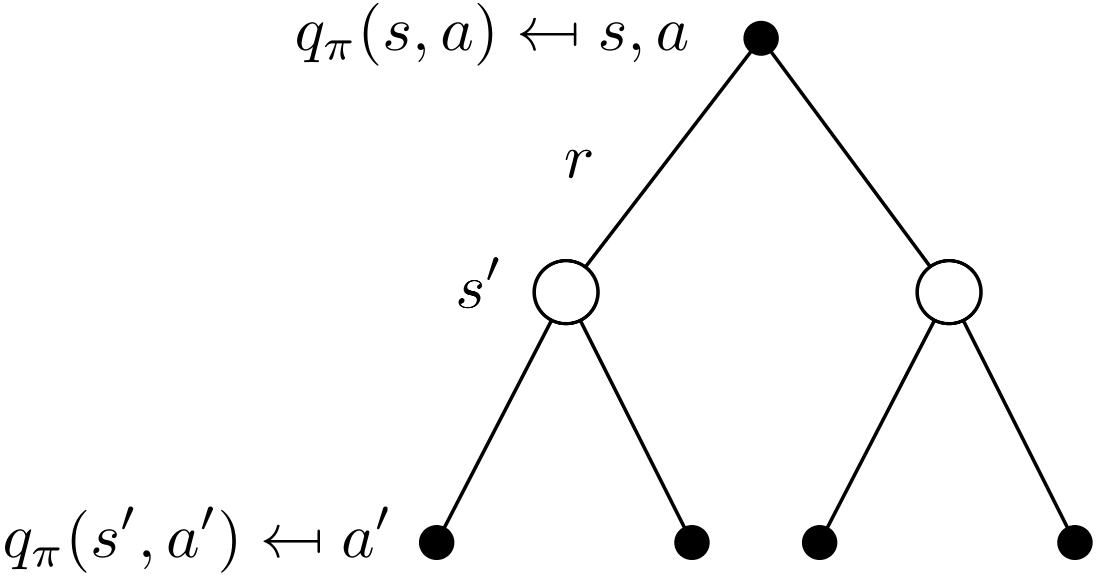
$$v_\pi(s) = \sum_a \pi(a|s)\left(R_s^a + \gamma \sum_{s'} P_{ss'}^a v_\pi(s')\right)$$

Action value function

(expected return from state s, taking action a, then following policy π)

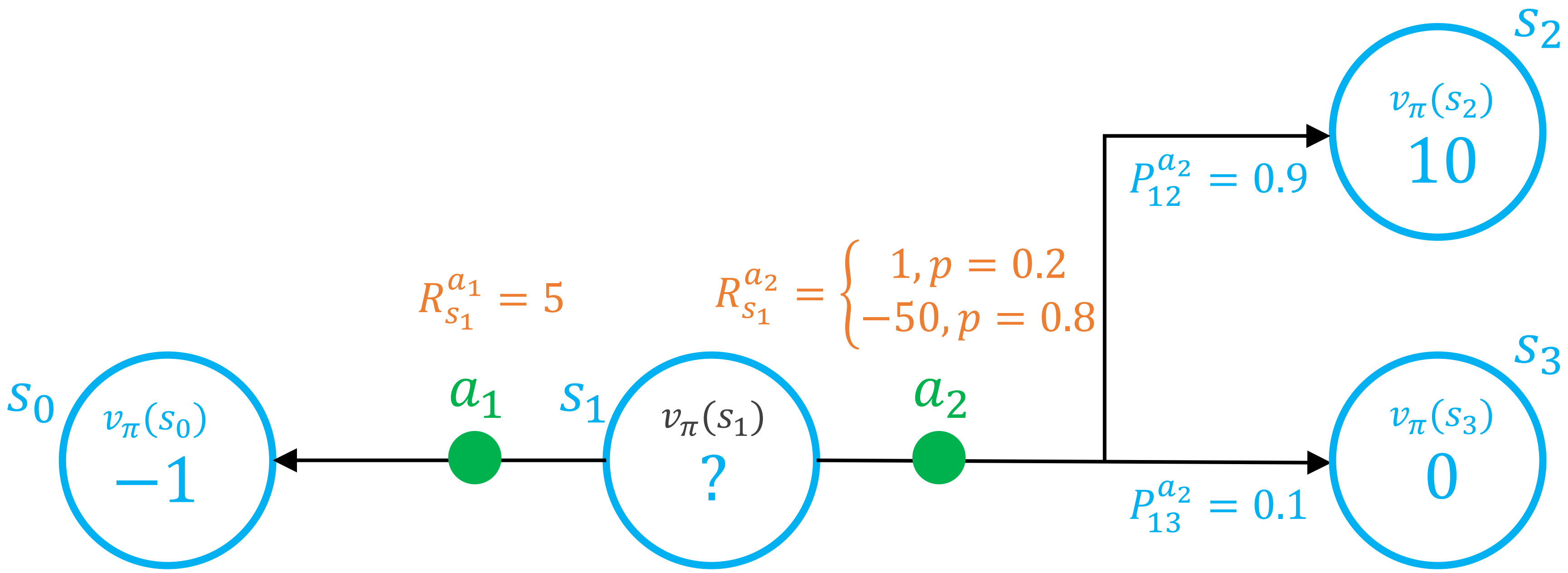
$$q_\pi(s,a) = E[G_t|s,a]$$

$$q_\pi(s,a) = E[R_s^a + \gamma q_\pi(s',a')|s,a]$$



$$q_\pi(s,a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a \sum_{a'} \pi(a'|s') q_\pi(s',a')$$

Example

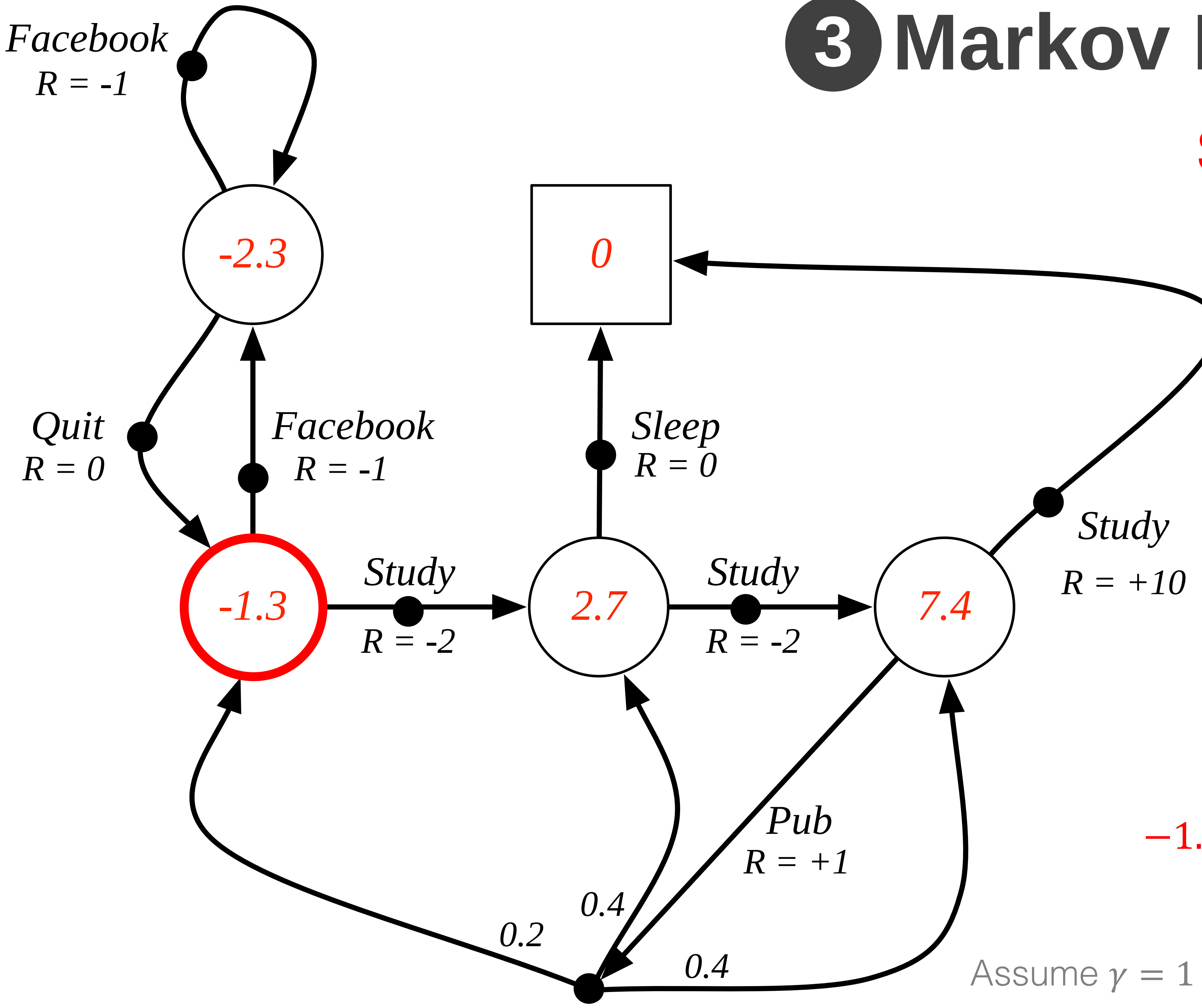


$$v_\pi(s) = \underbrace{\sum_a \pi(a|s)}\left(\underbrace{R_s^a} + \gamma \underbrace{\sum_{s'} P_{ss'}^a v_\pi(s')}\right)$$

$$v_\pi(s_1) = \overset{\pi(a_1|s_1)}{(0.5)}\underbrace{(\underset{R_{s_1}^{a_1}}{5} - \underset{v_\pi(s_0)}{1})}_{q_\pi(s_1,a_1)} + \overset{\pi(a_2|s_1)}{(0.5)}\underbrace{[\underset{R_{s_1}^{a_2}}{(0.2)(1) + (0.8)(-50)} + \overset{P_{12}^{a_2}}{(0.9)}\underset{v_\pi(s_2)}{(10)} + \overset{P_{13}^{a_2}}{(0.1)}\underset{v_\pi(s_3)}{(0)}]}_{q_\pi(s_1,a_2)} \qquad \gamma = 1$$

$$q_\pi(s,a) = E[R + \gamma v_\pi(s')]$$

3 Markov Decision Process



State value function, $v_\pi(s)$

Assumptions:

$$\pi(a|s) = \frac{1}{\#\text{ of reachable states}}$$

(randomly select an action)

$$\gamma = 1$$

(no discounting)

Bellman Expectation Equation

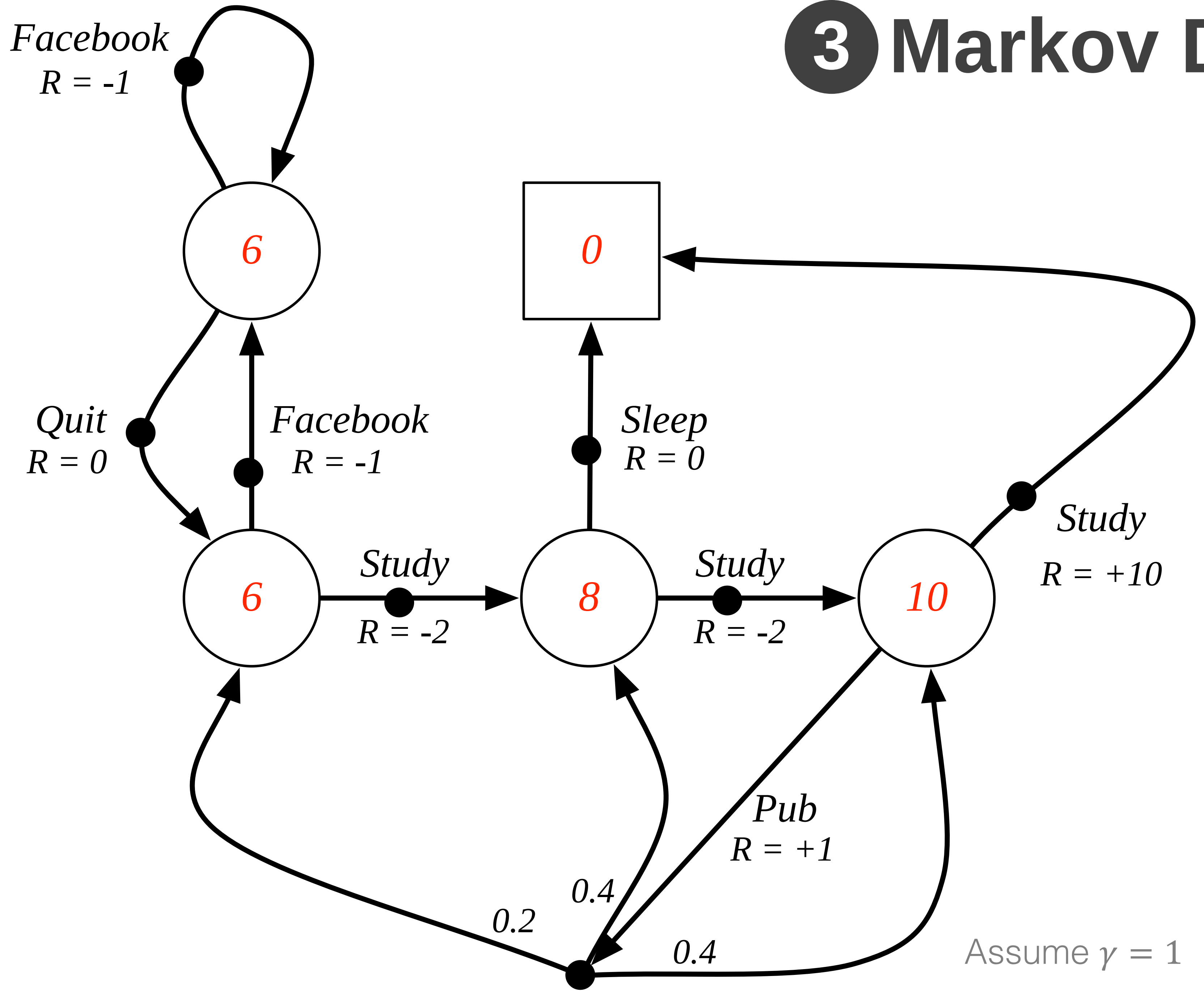
$$v(s) = E[R_s^a + \gamma v(s')|s]$$

$$-1.3 \approx 0.5\times(-1-2.3) + 0.5\times(-2+2.7)$$

Assume $\gamma = 1$

Example from David Silver, UCL, 2015

3 Markov Decision Process



Optimal **state-value** function, $v_*(s)$

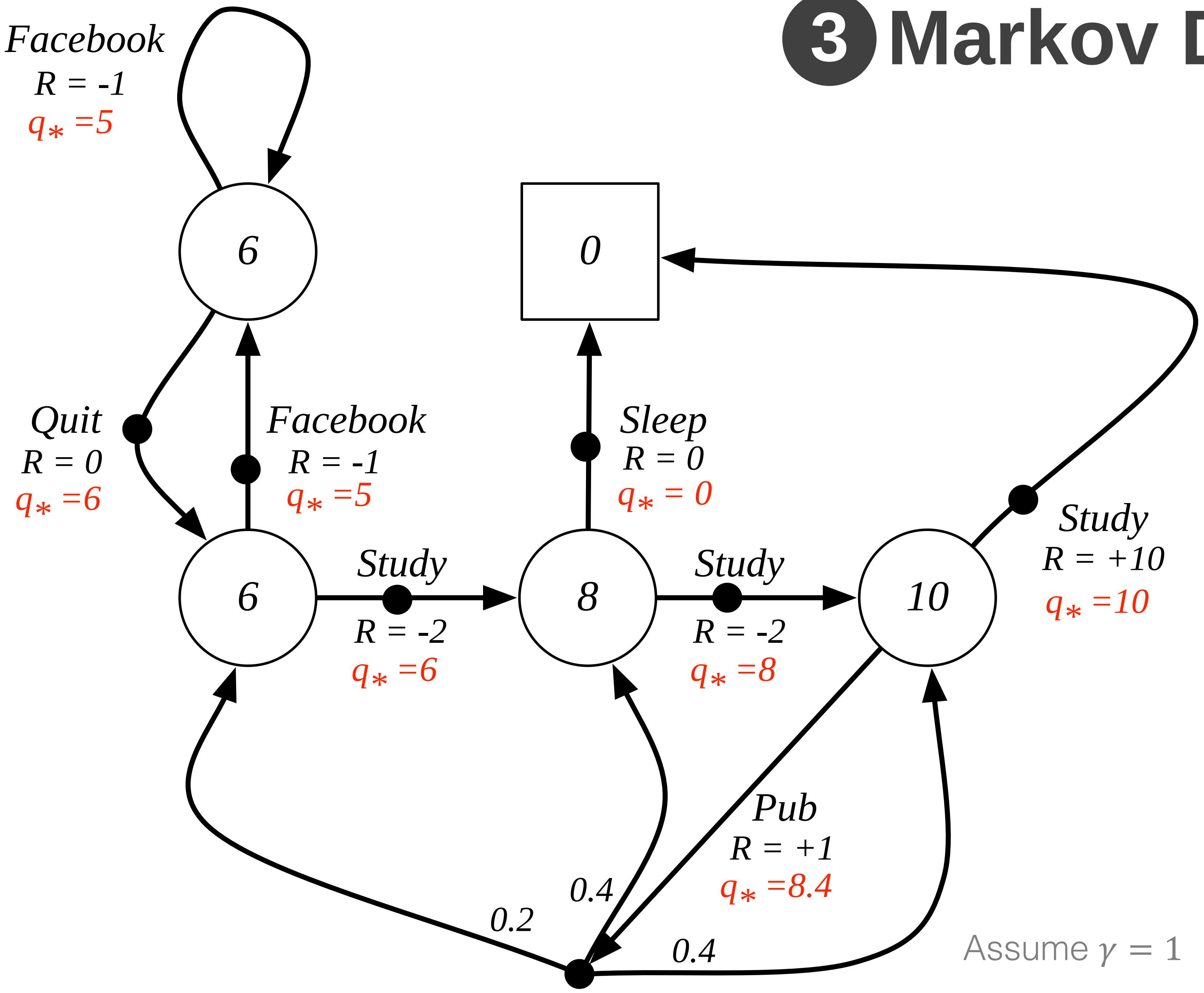
Maximum value function over all policies

$$v_*(s) = \max_{\pi} v_\pi(s)$$

Assume $\gamma = 1$

Example from David Silver, UCL, 2015

3 Markov Decision Process



Assume $\gamma = 1$

Optimal **state-value** function, $v_*(s)$

Maximum value function over all policies

$$v_*(s) = \max_{\pi} v_\pi(s)$$

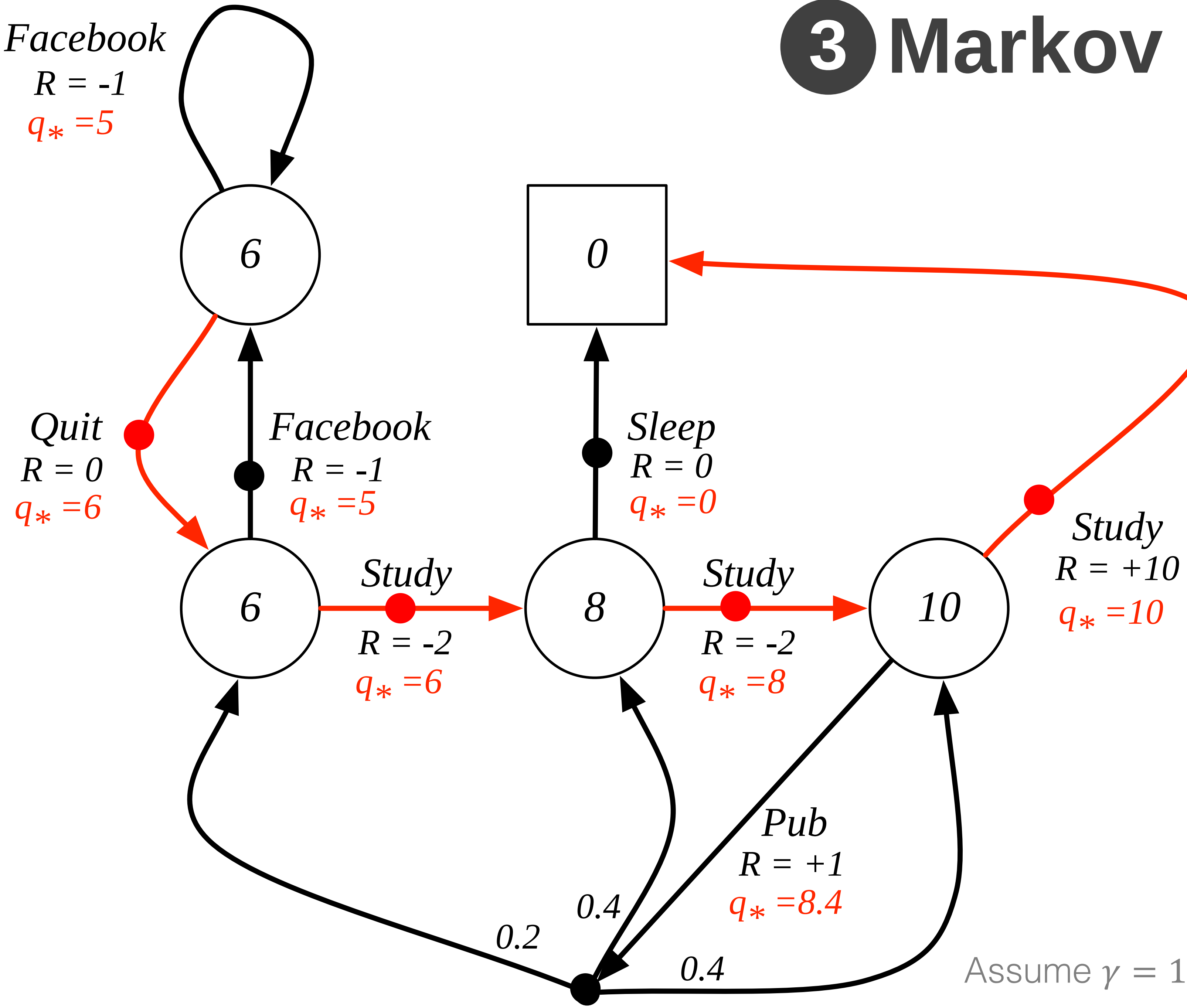
Optimal **action-value** function, $q_*(s, a)$

Maximum value function over all policies

$$q_*(s, a) = \max_{\pi} q_\pi(s, a)$$

Example from David Silver, UCL, 2015

❸ Markov Decision Process



Optimal **policy**, $\pi_*(s)$

Which action to take at each moment

$$\pi_*(s) = \arg\max_{a} q_*(s, a)$$

Once we have the optimal value functions, we've "solved" the MDP!

Assume $\gamma = 1$

Example from David Silver, UCL, 2015

Reinforcement Learning Roadmap

Knowledge of **Environment**

Perfect knowledge
Known Markov Decision Process

No knowledge
Must learn from experience

Dynamic Programming

What's a Markov Decision Process?
How do we find optimal policies?

Monte Carlo Control

How do we estimate our value functions?
How do we use the value functions to choose actions?