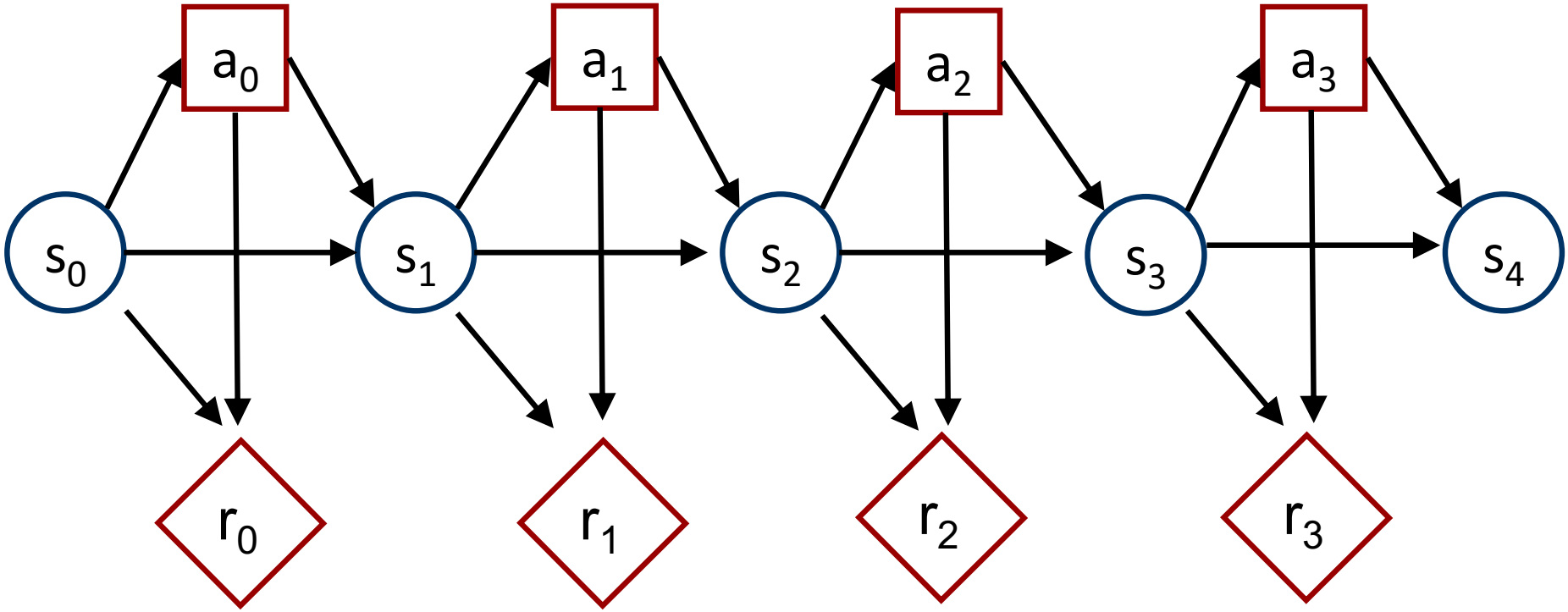
CS885 Reinforcement Learning Lecture 2a: May 4, 2018

Intro to Markov decision processes
[SutBar] Chap. 3, [Sze] Chap. 2,
[RusNor] Sec. 17.1-17.2, 17.4,
[Put] Chap. 2, 4, 5

Markov Decision Process

- Markov process augmented with…
  - Actions e.g., a_t
  - Rewards e.g., r_t



Current Assumptions

- Uncertainty: stochastic process
- Time: sequential process
- Observability: fully observable states
- No learning: complete model
- Variable type: discrete (e.g., discrete states and actions)

Rewards

- **Rewards**: $r_t \in \Re$
- **Reward function**: $R(s_t, a_t) = r_t$
  mapping from state-action pairs to rewards
- Common assumption: stationary reward function
  - $R(s_t, a_t)$ is the same $\forall t$
- Exception: terminal reward function often different
  - E.g., in a game: 0 reward at each turn and +1/-1 at the end for winning/losing
- Goal: **maximize sum of rewards** $\sum_t \boldsymbol{R(s_t, a_t)}$

Discounted/Average Rewards

- If process infinite, isn't $\sum_t R(s_t, a_t)$ infinite?

- Solution 1: discounted rewards
  - Discount factor: $0 \leq \gamma < 1$
  - Finite utility: $\sum_t \gamma^t R(s_t, a_t)$ is a geometric sum
  - γ induces an inflation rate of $1/\gamma - 1$
  - Intuition: prefer utility sooner than later

- Solution 2: average rewards
  - More complicated computationally
  - Beyond the scope of this course

Markov Decision Process

- Definition
  - Set of states: S
  - Set of actions: A
  - Transition model: $\Pr(s_t|s_{t-1}, a_{t-1})$
  - Reward model: $R(s_t, a_t)$
  - Discount factor: $0 \leq \gamma \leq 1$
    - discounted: $\gamma < 1$ undiscounted: $\gamma = 1$
  - Horizon (i.e., # of time steps): h
    - Finite horizon: $h \in \mathbb{N}$ infinite horizon: $h = \infty$

- Goal: find optimal policy

Inventory Management

- Markov Decision Process
  - States: inventory levels
  - Actions: {doNothing, orderWidgets}
  - Transition model: stochastic demand
  - Reward model: Sales – Costs - Storage
  - Discount factor: 0.999
  - Horizon: ∞

- Tradeoff: increasing supplies decreases odds of missed sales, but increases storage costs

Policy

- Choice of action at each time step

- Formally:
  - Mapping from states to actions
  - i.e., $\pi(s_t) = a_t$
  - Assumption: fully observable states
    - Allows a_t to be chosen only based on current state s_t

Policy Optimization

- Policy evaluation:
  - Compute expected utility

$$V^{\pi}(s_0) = \sum_{t=0}^{h} \gamma^t \sum_{s_t} \Pr(s_t|s_0, \pi)\, R(s_t, \pi(s_t))$$

- Optimal policy:
  - Policy with highest expected utility

$$V^{\pi^*}(s_0) \geq V^{\pi}(s_0) \;\; \forall \pi$$

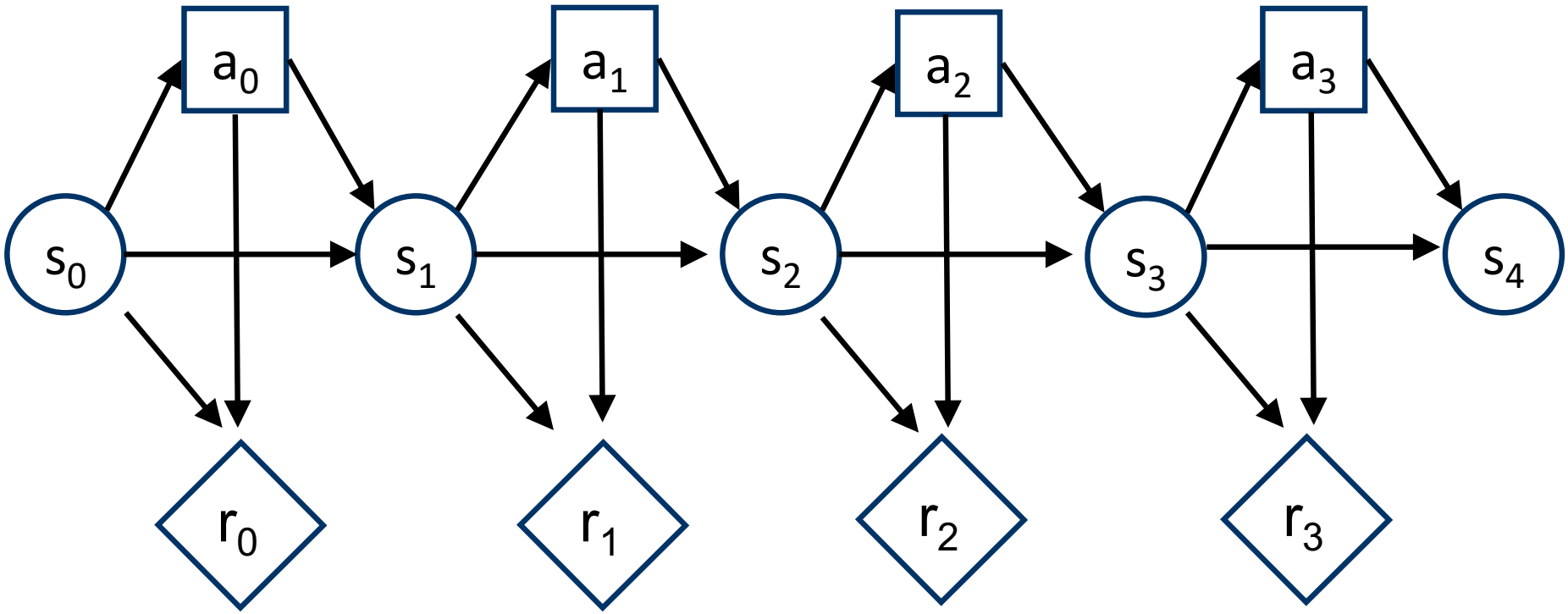
Policy Optimization

- Several classes of algorithms:
  - Value iteration
  - Policy iteration
  - Linear Programming
  - Search techniques

- Computation may be done
  - Offline: before the process starts
  - Online: as the process evolves

Value Iteration

- Performs dynamic programming
- Optimizes decisions in reverse order



Value Iteration

- Value when no time left:

$$V(s_h) = \max_{a_h} R(s_h, a_h)$$

- Value with one time step left:

$$V(s_{h-1}) = \max_{a_{h-1}} R(s_{h-1}, a_{h-1}) + \gamma \sum_{s_h} \Pr(s_h | s_{h-1}, a_{h-1}) V(s_h)$$

- Value with two time steps left:

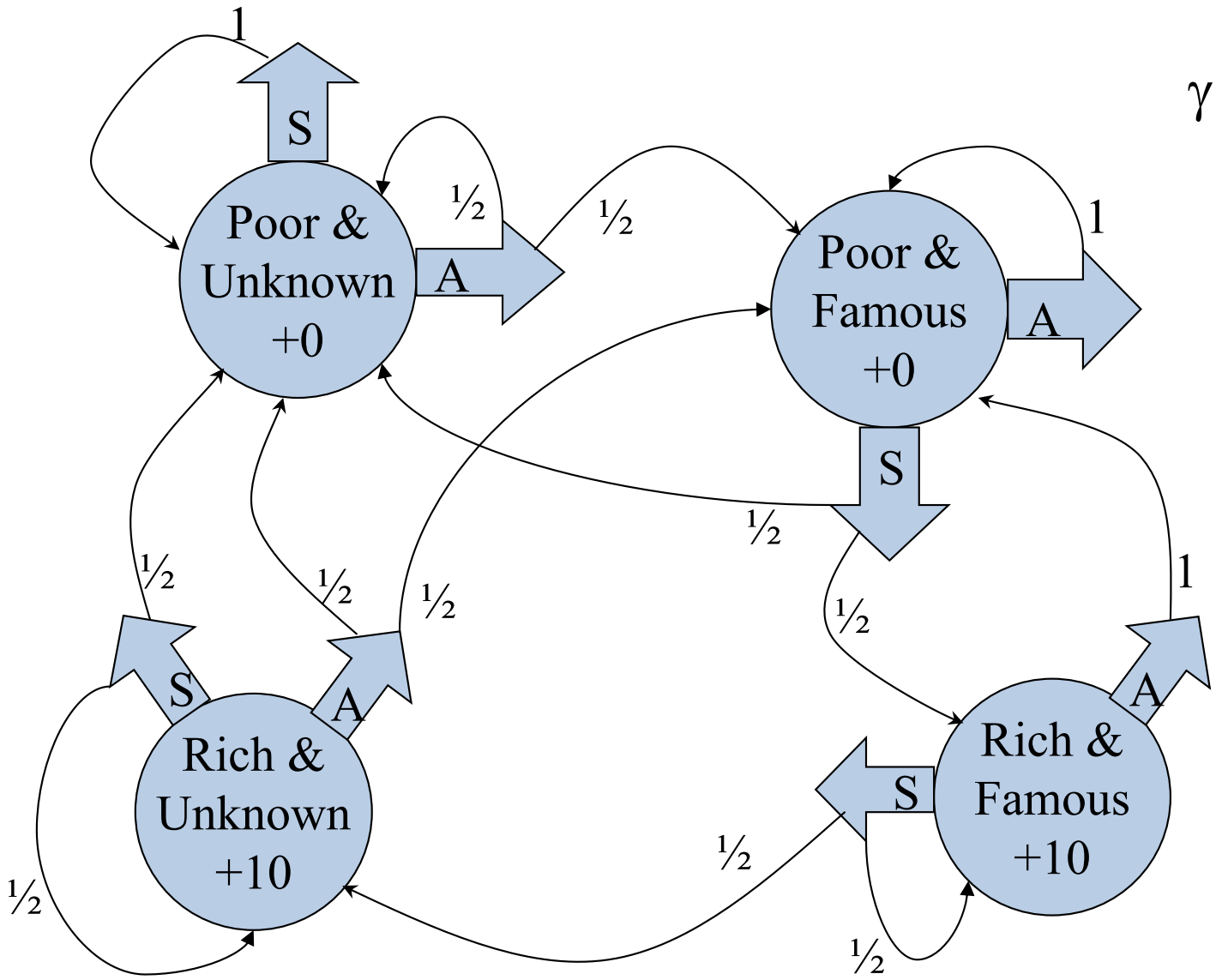
$$V(s_{h-2}) = \max_{a_{h-2}} R(s_{h-2}, a_{h-2}) + \gamma \sum_{s_{h-1}} \Pr(s_{h-1} | s_{h-2}, a_{h-2}) V(s_{h-1})$$

- …
- Bellman's equation:

$$V(s_t) = \max_{a_t} R(s_t, a_t) + \gamma \sum_{s_{t+1}} \Pr(s_{t+1} | s_t, a_t) V(s_{t+1})$$

$$a_t^* = \operatorname{argmax}_{a_t} R(s_t, a_t) + \gamma \sum_{s_{t+1}} \Pr(s_{t+1} | s_t, a_t) V(s_{t+1})$$

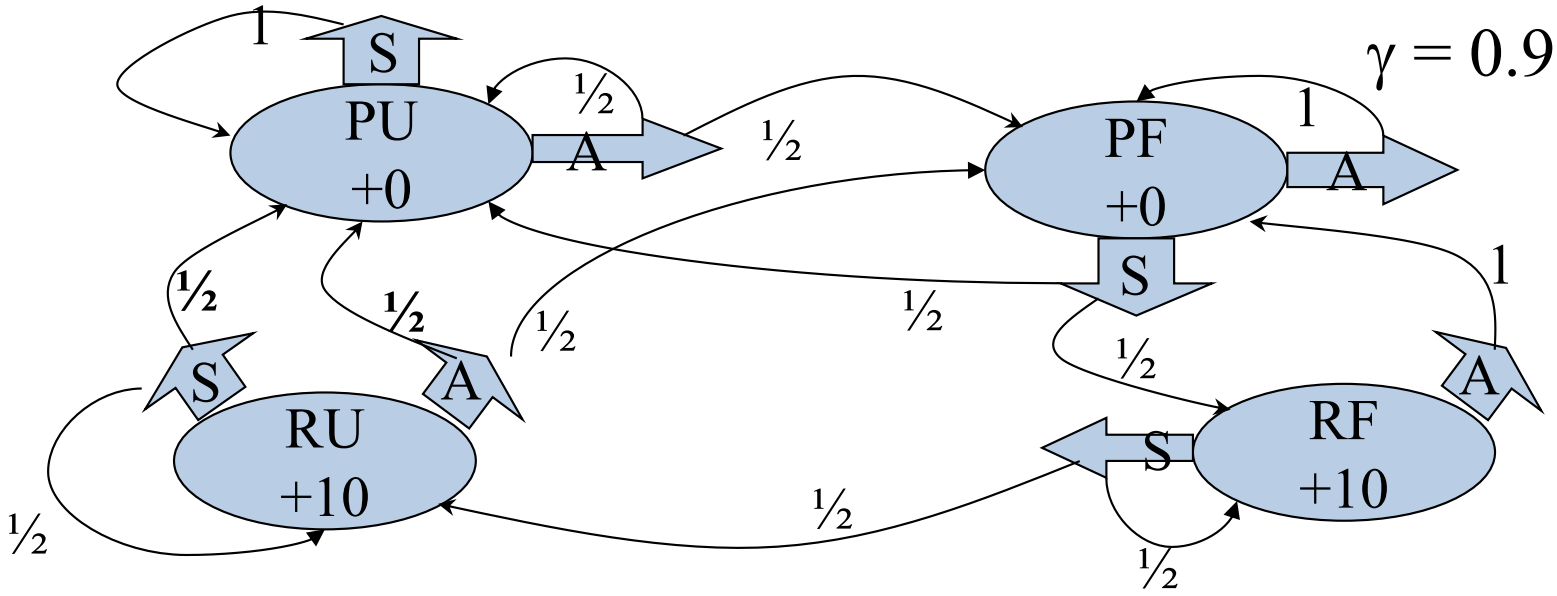
A Markov Decision Process



$\gamma = 0.9$

You own a company

In every state you must choose between **S**aving money or **A**dvertising



t	$V(PU)$	$\pi(PU)$	$V(PF)$	$\pi(PF)$	$V(RU)$	$\pi(RU)$	$V(RF)$	$\pi(RF)$
h	0	A,S	0	A,S	10	A,S	10	A,S
$h-1$	0	A,S	4.5	S	14.5	S	19	S
$h-2$	2.03	A	8.55	S	16.53	S	25.08	S
$h-3$	4.76	A	12.20	S	18.35	S	28.72	S
$h-4$	7.63	A	15.07	S	20.40	S	31.18	S
$h-5$	10.21	A	17.46	S	22.61	S	33.21	S

Finite Horizon

- When h is finite,
- Non-stationary optimal policy
- Best action different at each time step
- Intuition: best action varies with the amount of time left

Infinite Horizon

- When h is infinite,
- Stationary optimal policy
- Same best action at each time step
- Intuition: same (infinite) amount of time left at each time step, hence same best action

- Problem: value iteration does an infinite number of iterations…

Infinite Horizon

- Assuming a discount factor γ, after n time steps, rewards are scaled down by γ^n
- For large enough n, rewards become insignificant since $\gamma^n \to 0$
- Solution:
  - pick large enough n
  - run value iteration for n steps
  - Execute policy found at the n^{th} iteration