Lecture 17: Topics in Fairness

CIS 7000: Trustworthy Machine Learning

Spring 2024

Agenda

- Selective compliance
- Fairness in sequential decision-making

Fairness and Human-AI Systems

BUSINESS

Algorithms were supposed to make Virginia judges fairer. What happened was far more complicated.

Analysis by Andrew Van Dam
Staff writer | + Follow

November 19, 2019 at 7:00 a.m. EST

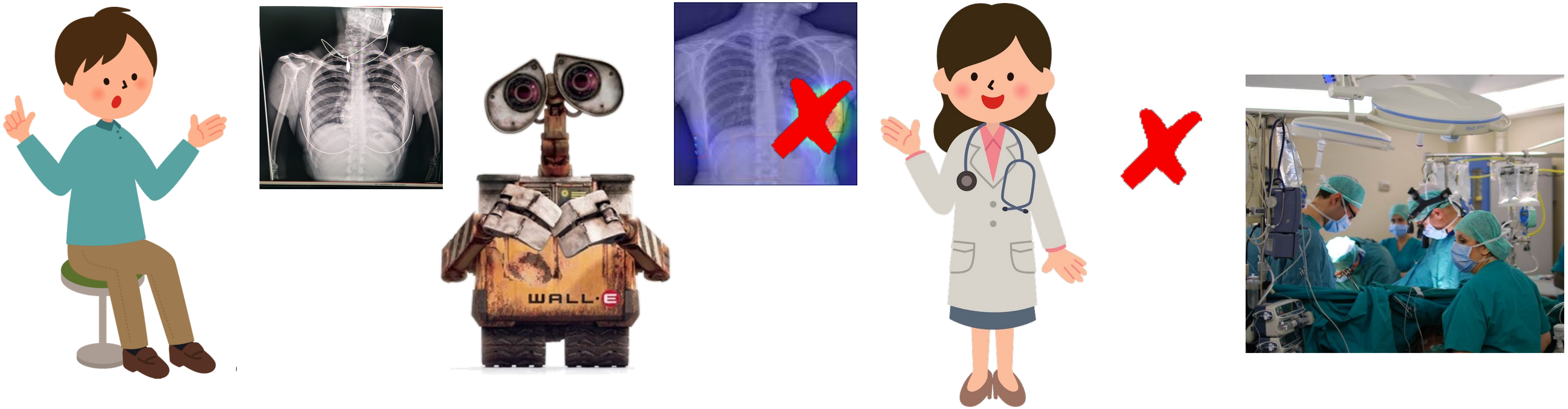
The Accomack County Courthouse in February of this year. (Timothy C. Wright for the Washington Post)

Share Comment 5 Save

We tend to assume the near-term future of automation will be built on man-machine partnerships. Our robot sidekicks will compensate for the squishy inefficiencies of the human brain, while human judgment will sand down their cold, mechanical edges.

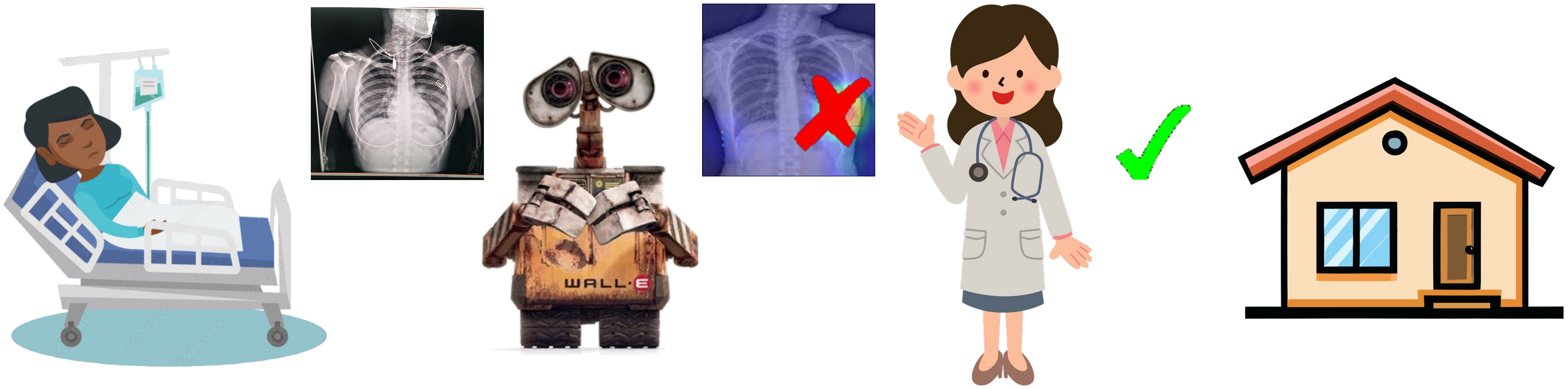
Selective Compliance

- Humans choose when to comply with algorithmic recommendations, with potentially problematic fairness consequences



Selective Compliance

- Humans choose when to comply with algorithmic recommendations, with potentially problematic fairness consequences

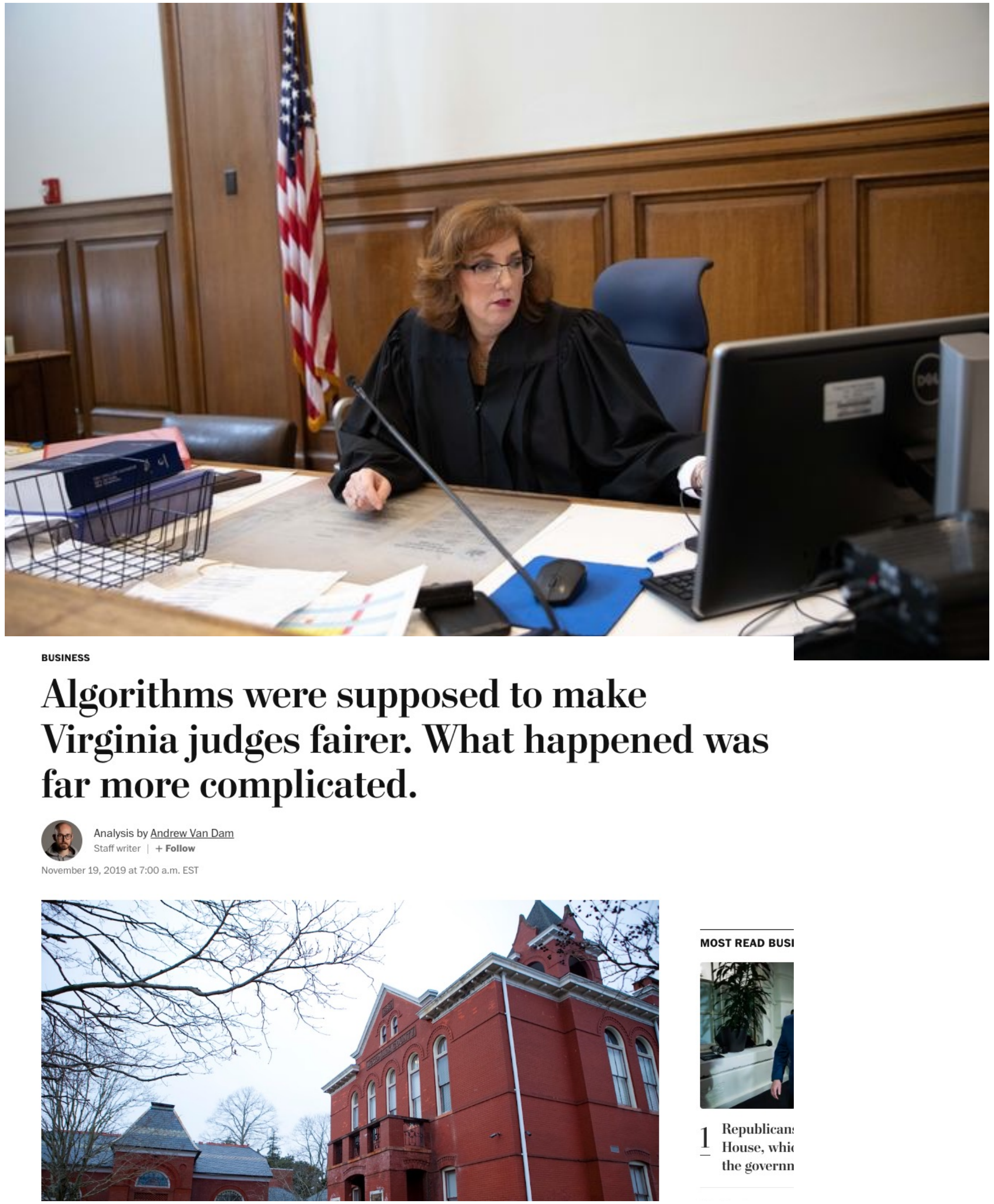


AI is fair

final decision is unfair

Failures

- Virginia sentencing data
- No benefit in safety or incarceration; racial disparities increased in courts where algorithm was used more
- Conflicting objectives, e.g., judges are lenient towards younger defendants
- Judges were more likely to sentence leniently for white defendants with high risk scores than for black defendants with the same score


BUSINESS

Algorithms were supposed to make Virginia judges fairer. What happened was far more complicated.

Analysis by Andrew Van Dam
Staff writer | + Follow
November 19, 2019 at 7:00 a.m. EST

Stevenson and Doleac, 2022; Van Dam, 2019

Formalizing Selective Compliance

- **Decision-making problem**
  - Finite number of types $x \in X = [k] = \{1, \ldots, k\}$
  - Binary indicator $a \in A = \{0, 1\}$ of protected attribute
  - Binary decision $y \in \{0, 1\}$ (e.g., should we treat the patient?)

- **Policy:** $\pi: X \times A \to [0,1]$ maps x to the prob of a decision of $y = 1$:

$$\hat{y} \sim \mathrm{Bernoulli}\big(\pi(x, a)\big)$$

Formalizing Selective Compliance

- **Human policy:** π_H used by human in absence of AI
- **AI policy:** π_A is the AI recommendation
- **Compliance function:** mapping $c: X \times A \rightarrow \{0,1\}$, indicating whether the human complies with the AI recommendation for (x, a)
- **Human-AI collaborative policy:**

$$\pi_C(x,a) = \begin{cases} \pi_A(x,a) & \text{if } c(x,a) = 1 \\ \pi_H(x,a) & \text{otherwise} \end{cases}$$

Equality of Opportunity

- **Recall:** $\Pr(\,\hat{y} = 1 \mid y = 1, a = 0\,) = \Pr(\,\hat{y} = 1 \mid y = 1, a = 1\,)$

- Let the average score for subgroup a be

$$\bar{\pi}(a) = \sum_{x \in X} \pi(x, a) \cdot P(\,x \mid a, y = 1\,)$$

- Policy π is fair if and only if

$$\alpha(\pi) \coloneqq |\bar{\pi}(1) - \bar{\pi}(0)| = 0$$

- WLOG assume $\bar{\pi}_H(1) \geq \bar{\pi}_H(0)$

Compliance-Robust Fairness

- **Goal:** An algorithmic policy π_A that **never reduces fairness** regardless of how the end user chooses to comply
  - Then, π_C is at least as fair as π_H for **any** compliance fn c
  - For any c, we have $\alpha(\pi_C) \leq \alpha(\pi_H)$

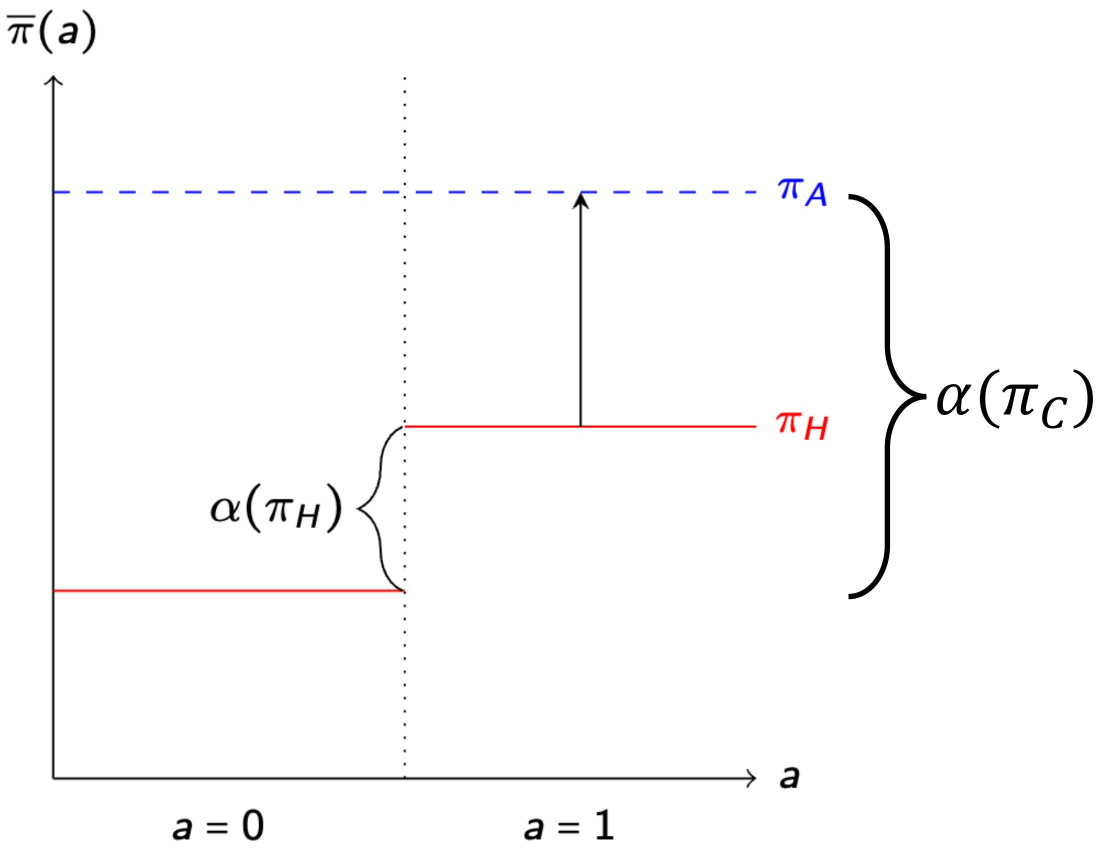
- **Note:**
  - Cannot guarantee π_C is strictly fairer than π_H, since human can choose $c = 0$
  - Assume knowledge of π_H but nothing about c

Intuitive Example

- Suppose $\bar{\pi}_A(a) > \bar{\pi}_H(a)$ for all a

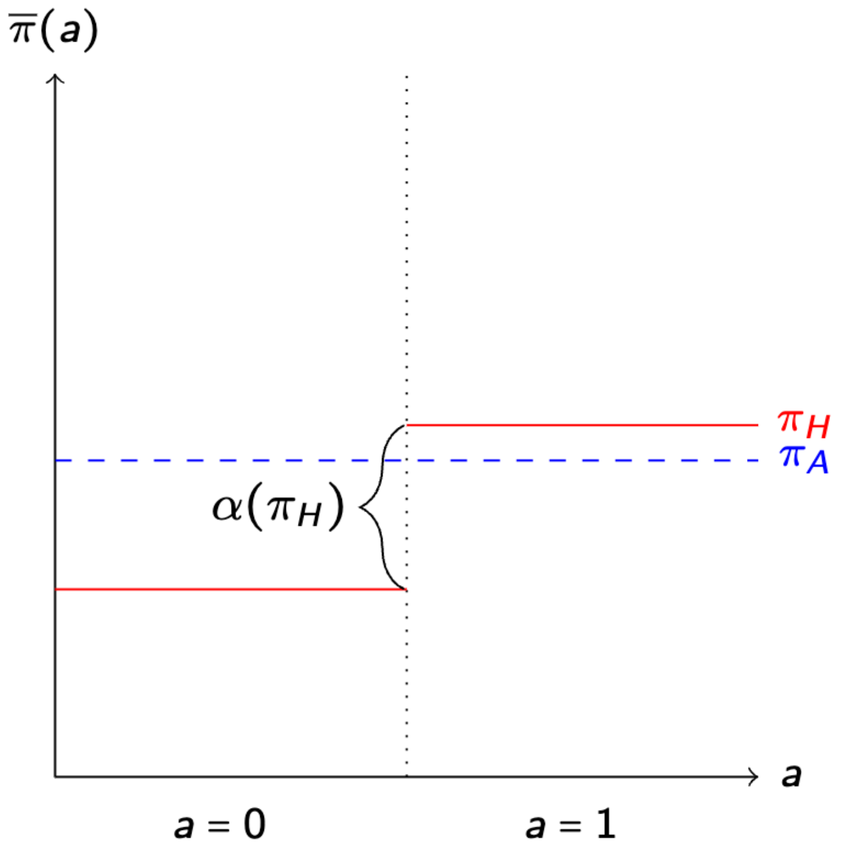
- Fairness decreases if human selectively complies for members of the subgroup a for which $\bar{\pi}_H(a) > \bar{\pi}_H(1-a)$

- Similar if $\bar{\pi}_A(a) < \bar{\pi}_H(a)$ for all a



Intuitive Example

- Modify π_A to be "sandwiched" between levels of π_H
- If $\bar{\pi}_H(0) < \bar{\pi}_A(a) < \bar{\pi}_H(1)$ for all $a \in A$, then π_A is compliance-robust!



Characterizing Compliance-Robustness

- **Assumption:** $P(x, a, y) > 0$ for all $x \in X$, $a \in A$, $y \in Y$

- **Theorem:** A policy π is compliance-robustly fair iff it satisfies:
  1. $\alpha(\pi) \leq \alpha(\pi_H)$
  2. $\pi_H(x, 0) \leq \pi(x, 0)$ for all $x \in X$
  3. $\pi_H(x, 1) \geq \pi(x, 1)$ for all $x \in X$

- Intuitively, no matter how the human complies, they can only reduce $\bar{\pi}_H(1)$ or increase $\bar{\pi}_H(0)$ (without “crossing” them)

Characterizing Compliance-Robustness

- **Corollary:** If $\alpha(\pi_H) = 0$, then the only compliance-robustly fair algorithmic policy is $\pi_A = \pi_H$
- If the human is perfectly fair, then any nontrivial algorithm can reduce fairness

What About Performance?

- Consider a loss function $\ell\colon [0,1] \times Y \mapsto \mathbb{R}$
  - Policy loss $L(\pi) = \mathbb{E}[\ell(\pi(x,a), y^*)]$

- Optimal policy: $\pi_* = \arg\min_\pi L(\pi)$
  - May not be compliance robustly fair (or fair in the traditional sense)

- **Assumption:** If for all $x \in X$ and $a \in A$, $\pi'(x,a) \leq \pi(x,a) < \pi^*(x,a)$ or $\pi'(x,a) > \pi(x,a) \geq \pi^*(x,a)$, then, $L(\pi) < L(\pi')$
  - Intuitively, if π' deviates further from π^* than π for all inputs, then π' has higher expected loss
  - Satisfied by common loss functions (mean squared error, mean absolute error, cross entropy, etc.)

Optimizing for Performance

- Optimization problem to compute the performance-maximizing compliance-robustly fair policy:

$$
\begin{aligned}
\pi_0 = \arg\min_{\pi} L(\pi) & & \\
\text{subj. to} \qquad & \alpha(\pi) \leq \alpha(\pi_H) & \\
& \pi_H(x,0) \leq \pi(x,0) & (\forall x \in \mathcal{X}) \\
& \pi(x,1) \leq \pi_H(x,0) & (\forall x \in \mathcal{X})
\end{aligned}
$$

- **Question:** Does a (nontrivial) solution always exist?

Existence of a Solution

- Consider the policy

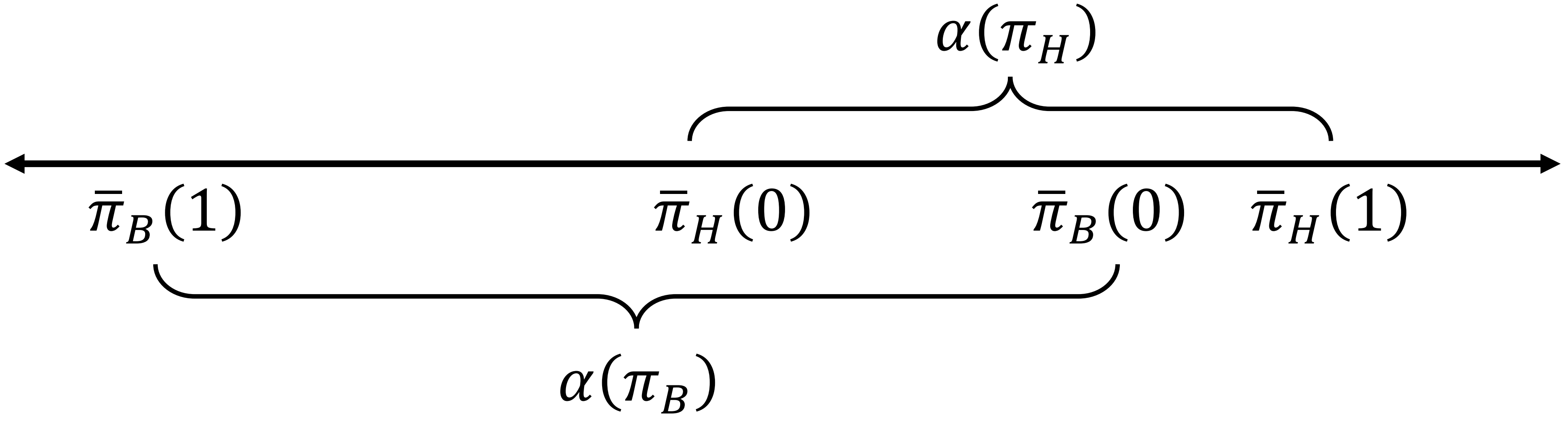
$$\pi_B(x,a) = \begin{cases} \max\{\pi_*(x,a), \pi_H(x,a)\} & \text{if } a = 0 \\ \min\{\pi_*(x,a), \pi_H(x,a)\} & \text{if } a = 1 \end{cases}$$

- **Intuition:** Tries to satisfy the constraints in our theorem
  2. $\pi_H(x,0) \leq \pi(x,0)$ for all $x \in X$
  3. $\pi_H(x,1) \geq \pi(x,1)$ for all $x \in X$

- But may not satisfy the first constraint!
  1. $\alpha(\pi) \leq \alpha(\pi_H)$

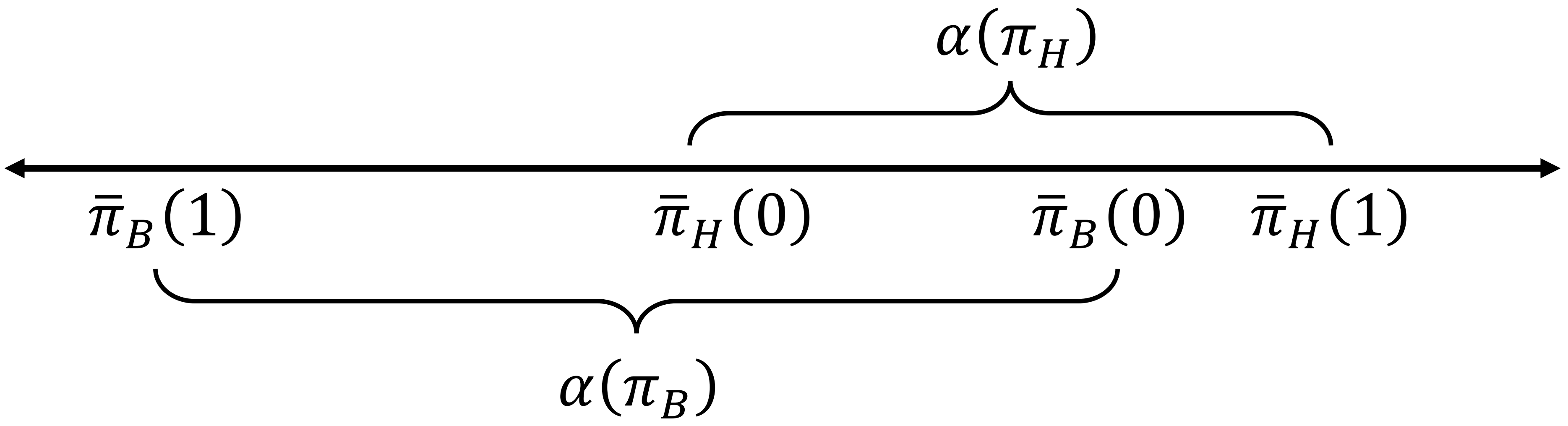
Existence of a Solution

- **Theorem:** If $\alpha(\pi_B) \leq \alpha(\pi_H)$, then π_B is compliance-robustly fair
- Why might $\alpha(\pi_B) \leq \alpha(\pi_H)$ fail to hold?
- **Intuition:** π_B may be unfair in the "opposite direction"



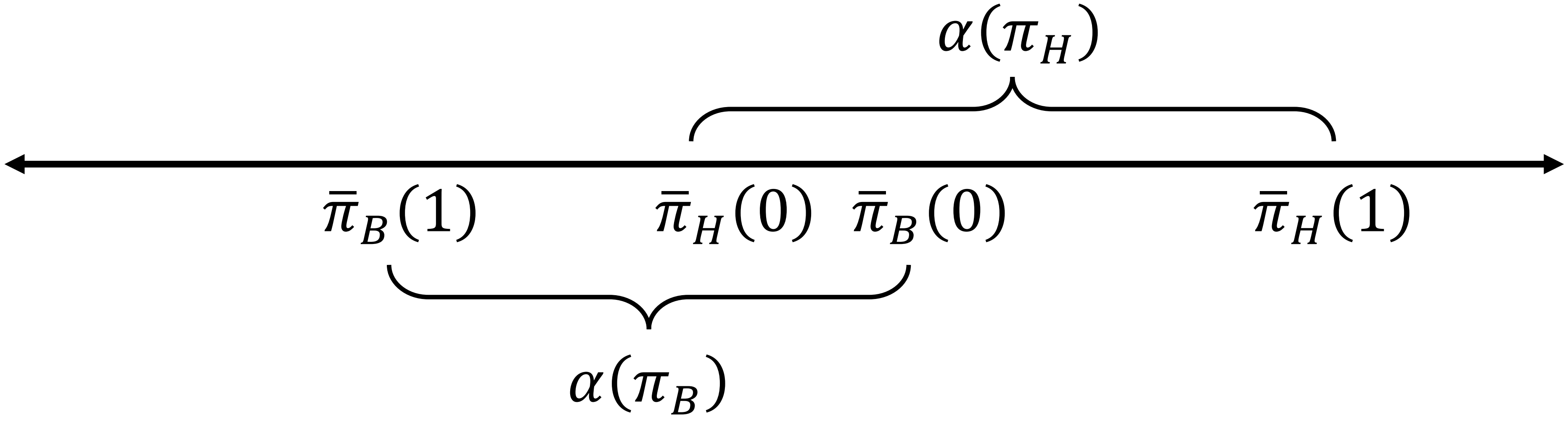
Existence of a Solution

- **Theorem:** If $\alpha(\pi_B) \leq \alpha(\pi_H)$, then π_B is compliance-robustly fair
- Why might $\alpha(\pi_B) \leq \alpha(\pi_H)$ fail to hold?
- **Solution:** Scale π_B back towards π_H



Existence of a Solution

- **Theorem:** If $\alpha(\pi_B) \leq \alpha(\pi_H)$, then π_B is compliance-robustly fair
- Why might $\alpha(\pi_B) \leq \alpha(\pi_H)$ fail to hold?
- **Solution:** Scale π_B back towards π_H



Existence of a Solution

- **Assumption:** $\alpha(\pi_H) \neq 0$ and $\pi_B \neq \pi_H$
- **Theorem:** There exists a performance-improving compliance-robustly fair policy π_0 if and only if the above assumption holds
- **Proof:** Above argument + intermediate value theorem

Tension with Traditional Fairness

- May not always be able to achieve compliance-robust fairness, performance improvement, and traditional fairness

- **Proposition:** There are settings where no traditionally fair policy is compliance-robustly fair while improving performance
  - **Intuition:** If π_* is very unfair, and π_H is close to π_*, then any traditionally fair policy cannot satisfy our constraints

- **Theorem:** If $\alpha(\pi_*) = 0$, then there exists a performance-improving policy that is both traditionally and compliance-robustly fair

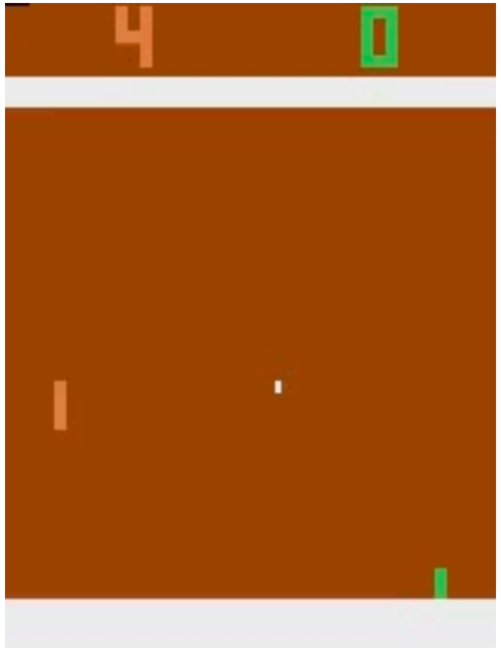
Summary

- Fairness for human-AI collaboration looks very different from traditional fairness
  - May need to forgo traditional fairness to improve end-to-end outcomes!
- New algorithms are needed to ensure fairness of final decisions

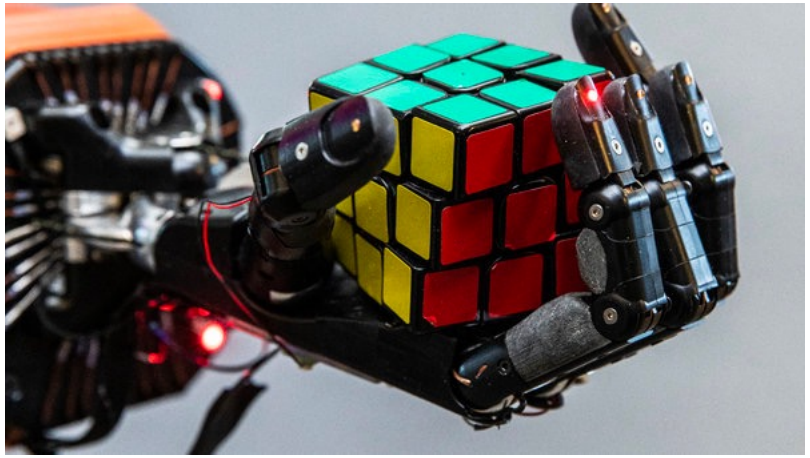
Agenda

- Selective compliance
- Fairness in sequential decision-making

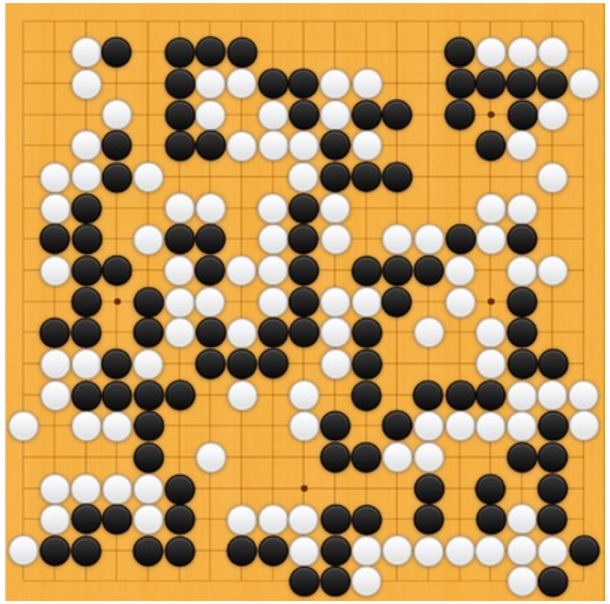
Reinforcement Learning



Atari

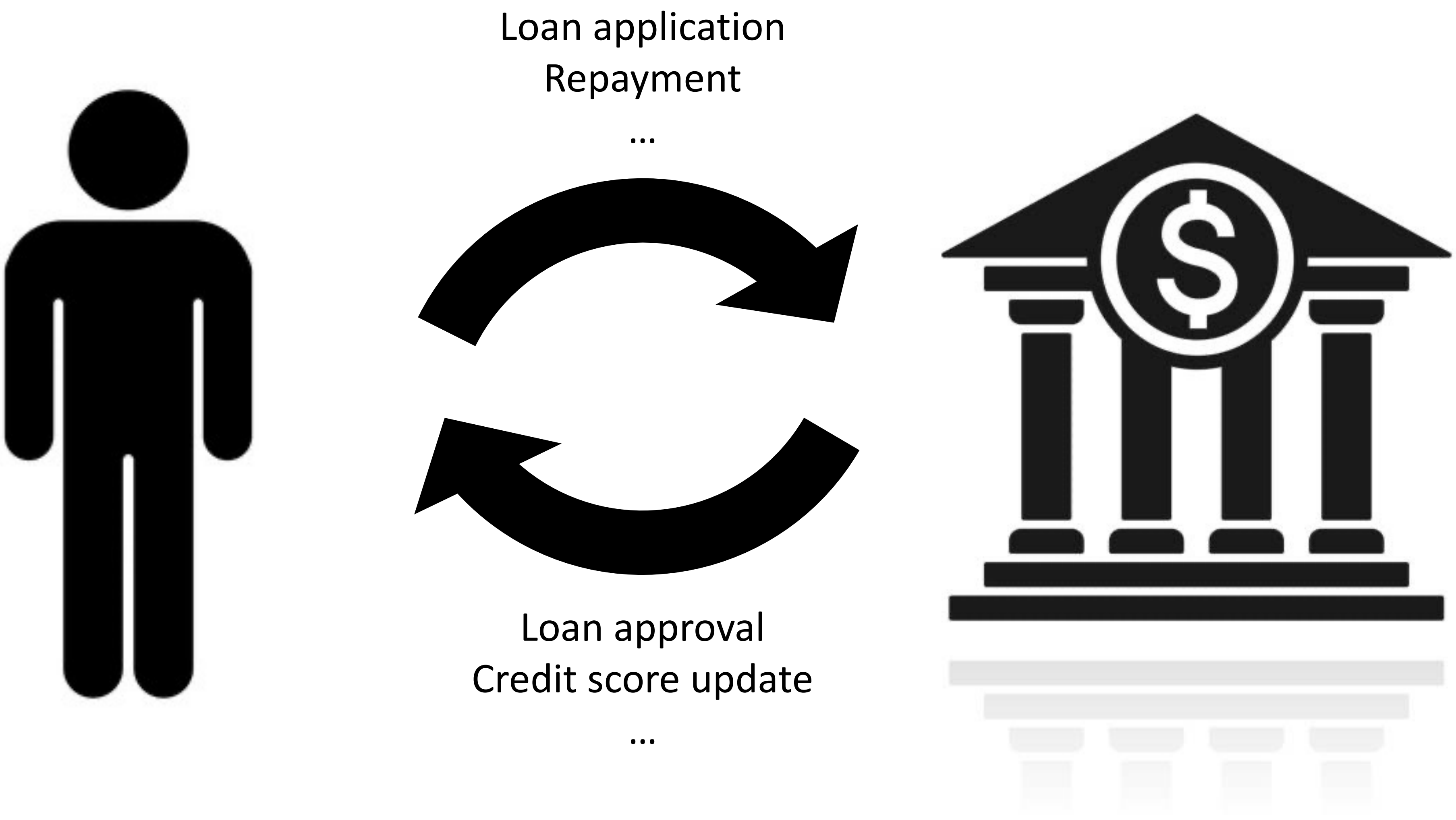


Robotics



AlphaGo

Real World Reinforcement Learning



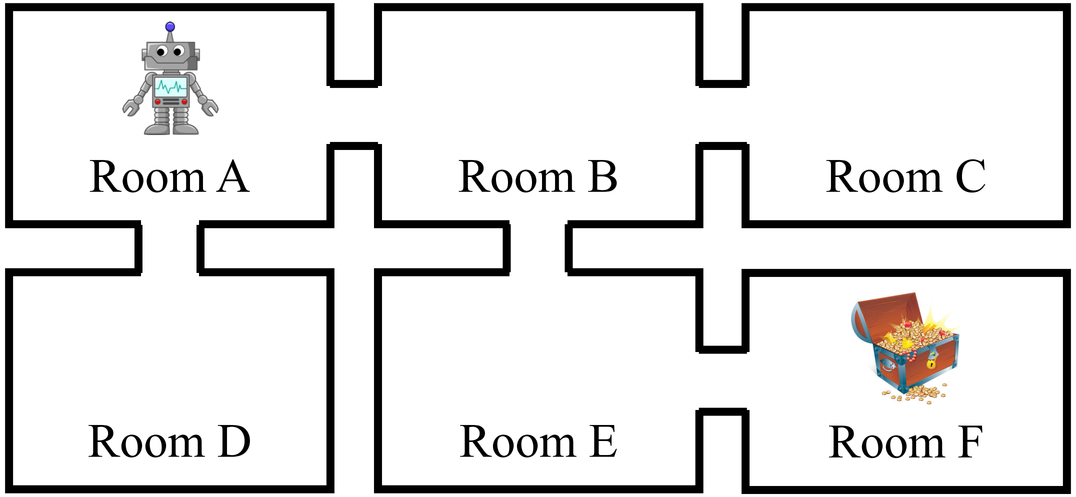
Real World Reinforcement Learning

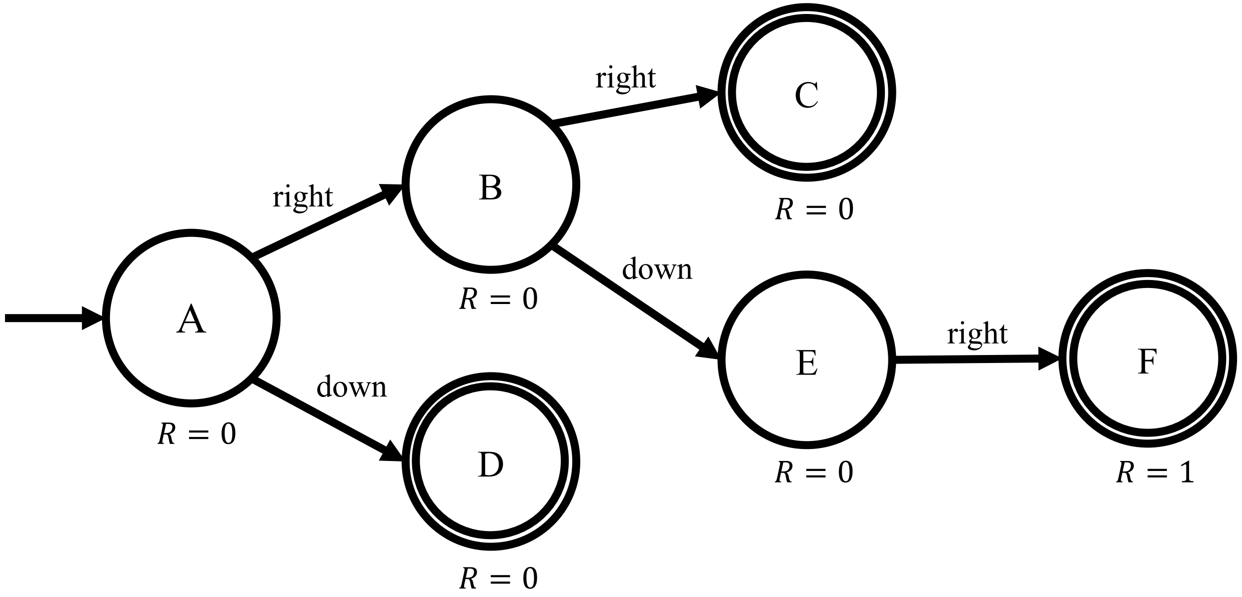
- Lots of systems involve sequential decisions
  - Banking/financial decision-making
  - Judicial decision-making
  - Medical decision-making
  - Education

- These systems should satisfy fairness for long-term outcomes

Markov Decision Processes (MDPs)

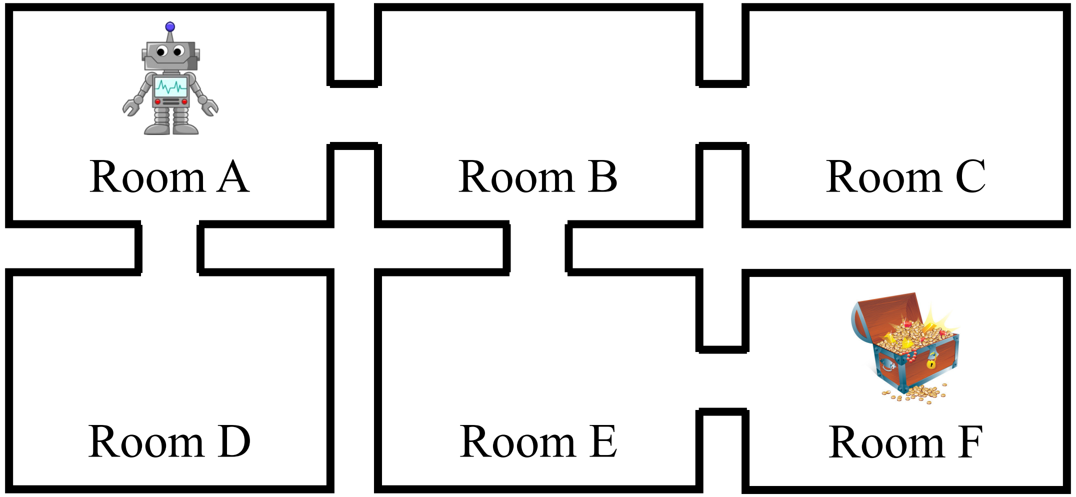
- **States** S
  - Set of rooms
- **Initial state** s_0
  - Room A
- **Actions** A
  - Direction to go
- **Transitions** $P: S \times A \to S$
  - Room that the robot ends up in
- **Rewards** $R: S \times A \to \mathbb{R}$
  - 1 for room F, 0 otherwise
- **Time horizon** T
  - How many steps the robot can take

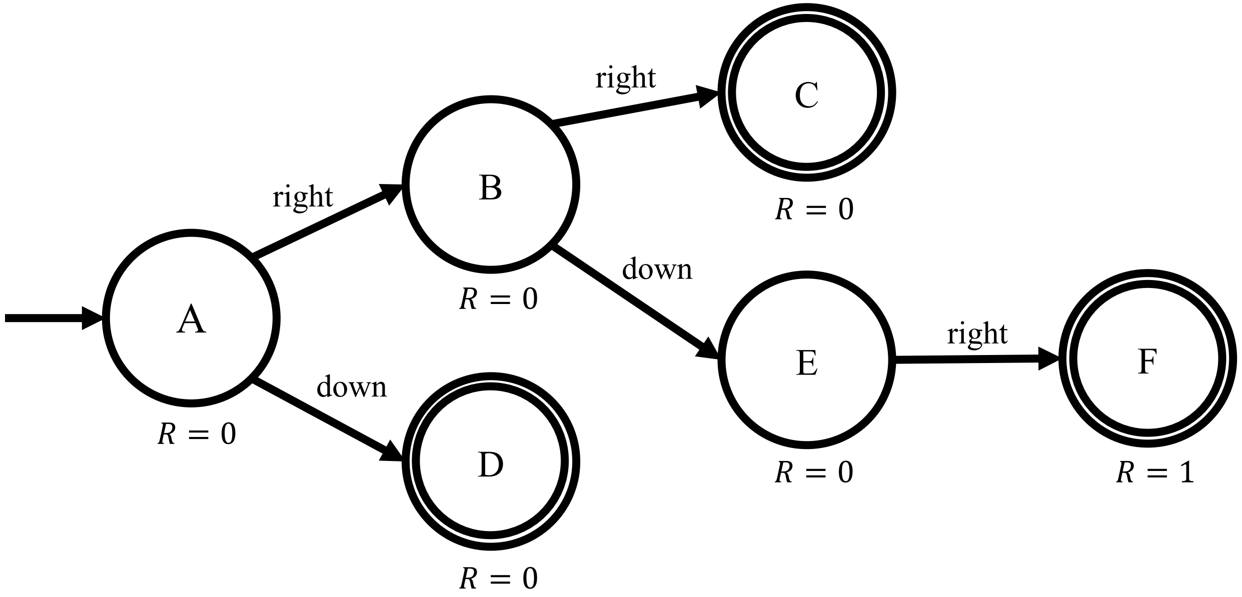




Markov Decision Processes (MDPs)

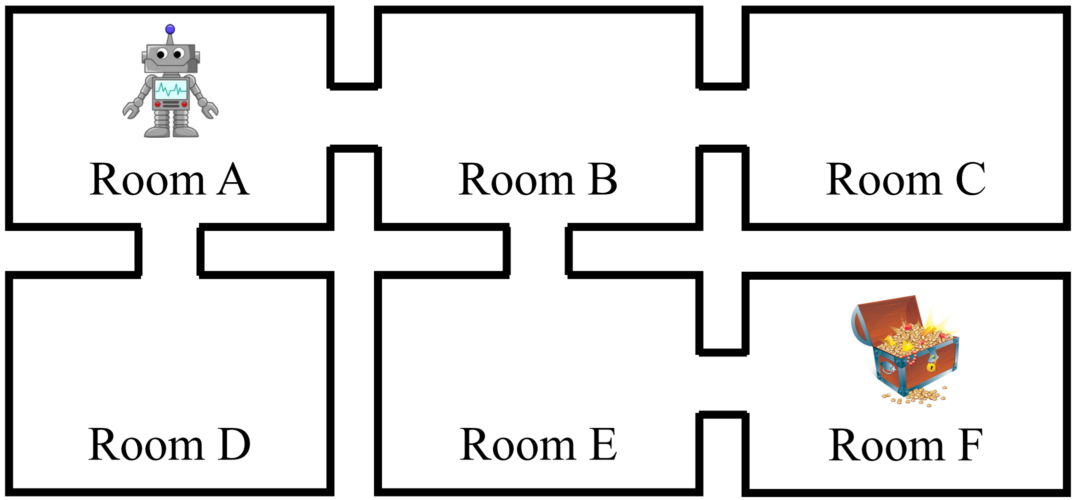
- **Policy** $\pi: S \rightarrow A$
  - Maps room to direction to take

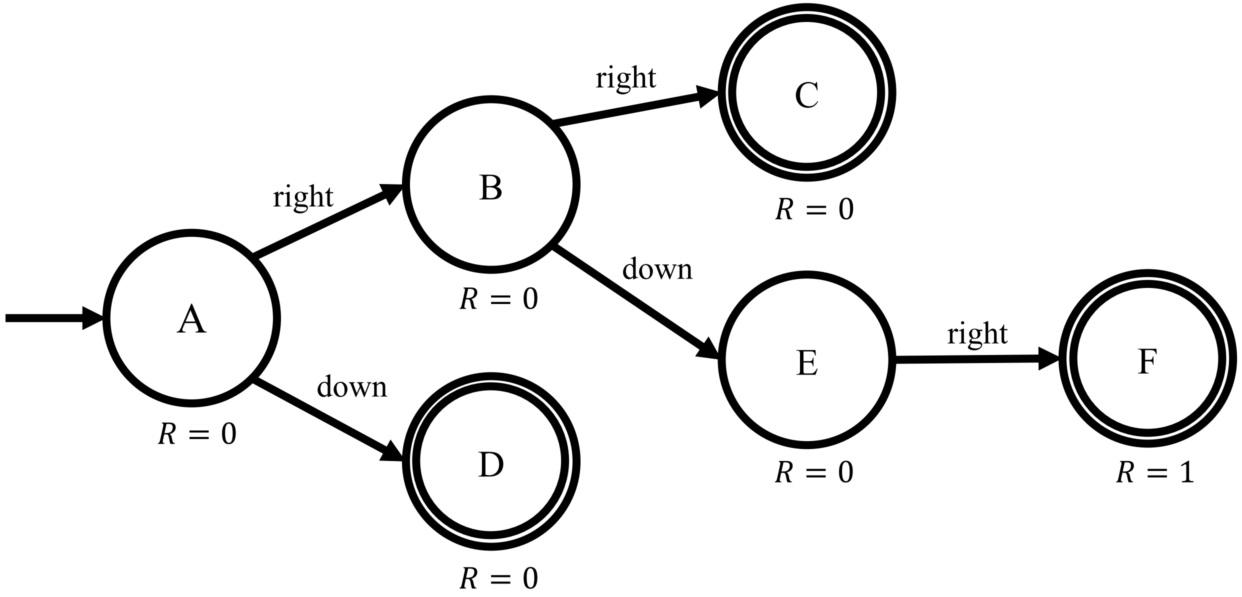




Markov Decision Processes (MDPs)

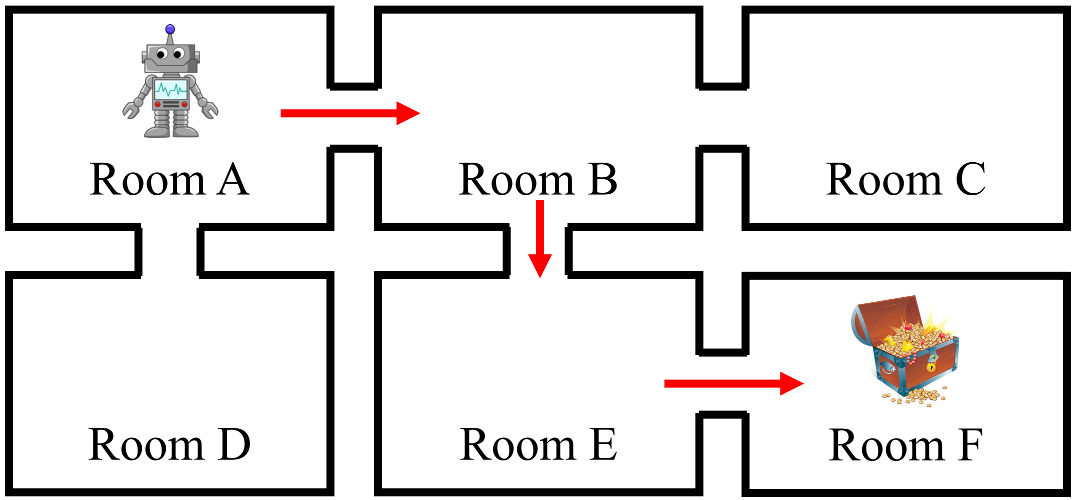
- **Policy** $\pi: [T] \times S \to A$
  - Maps room to direction to take

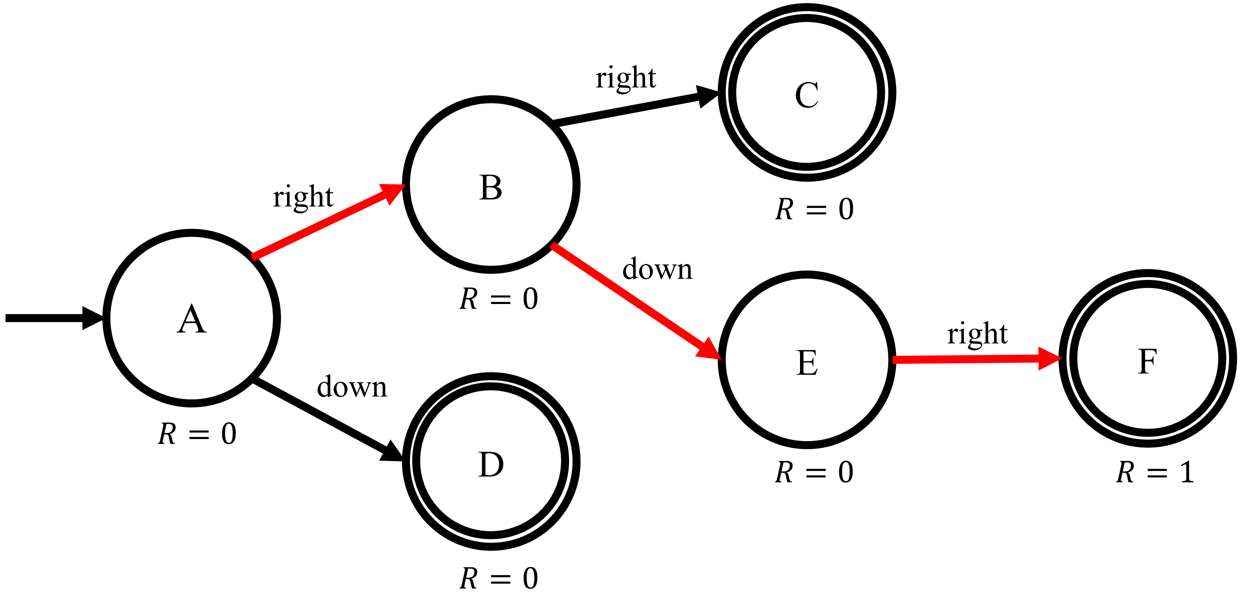




Markov Decision Processes (MDPs)

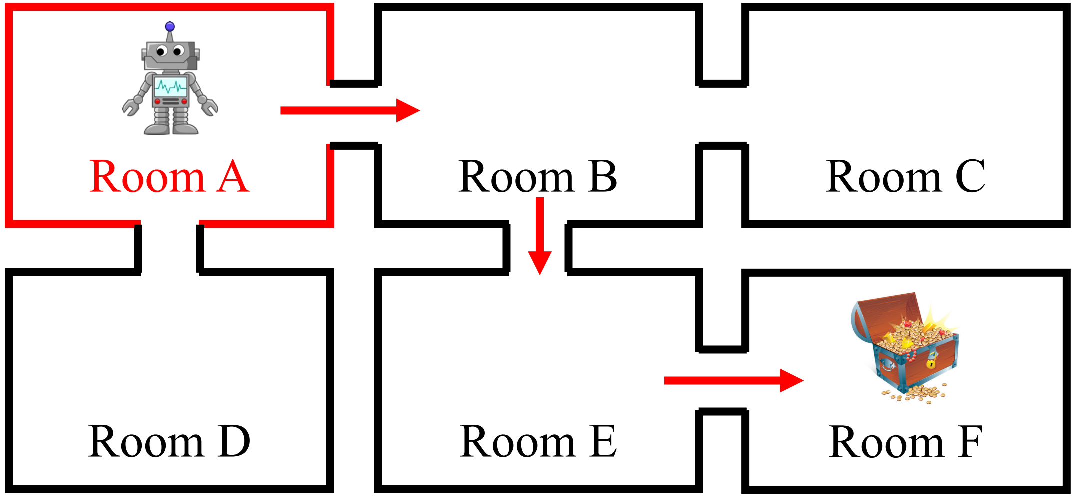
- **Policy** $\pi : [T] \times S \to A$
  - Maps room to direction to take

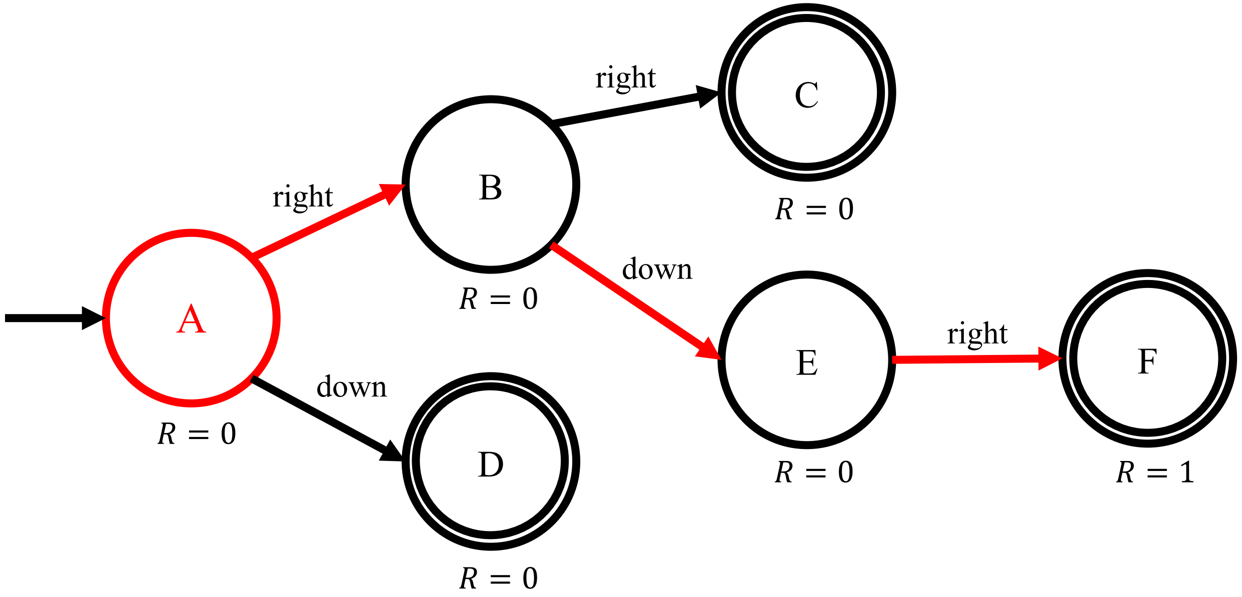




Markov Decision Processes (MDPs)

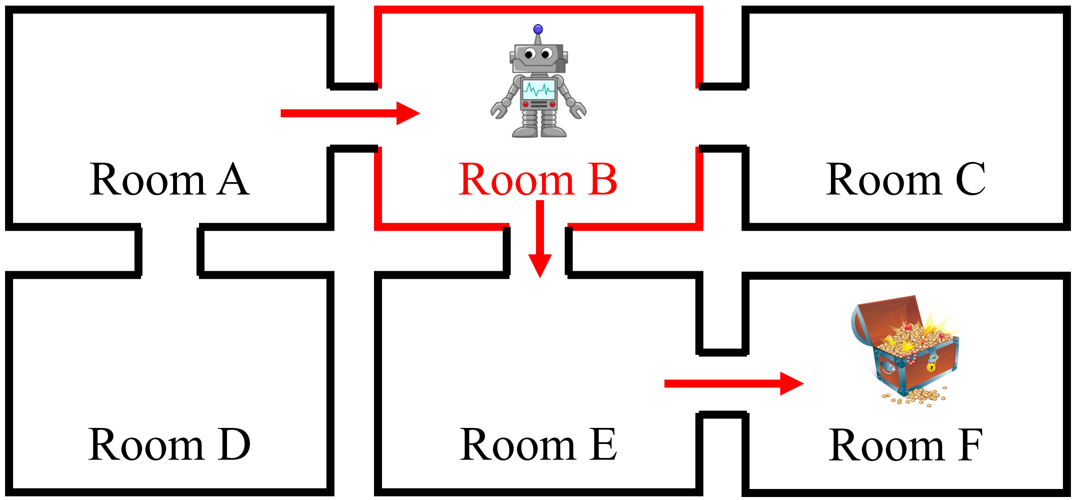
- **Policy** $\pi: [T] \times S \to A$
  - Maps room to direction to take

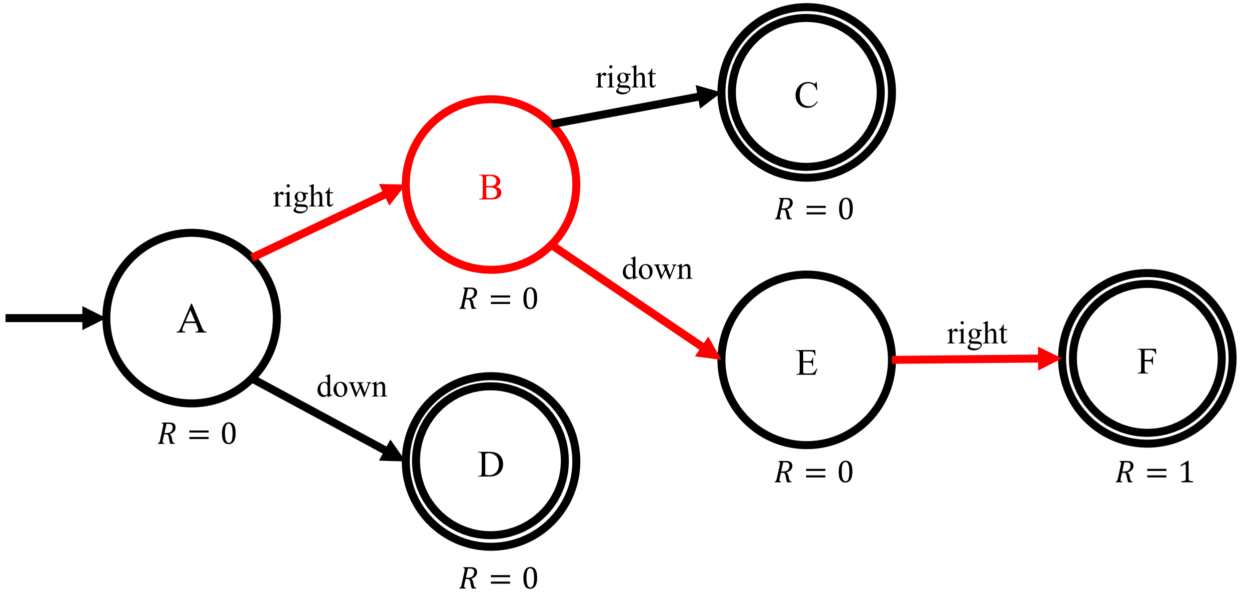




Markov Decision Processes (MDPs)

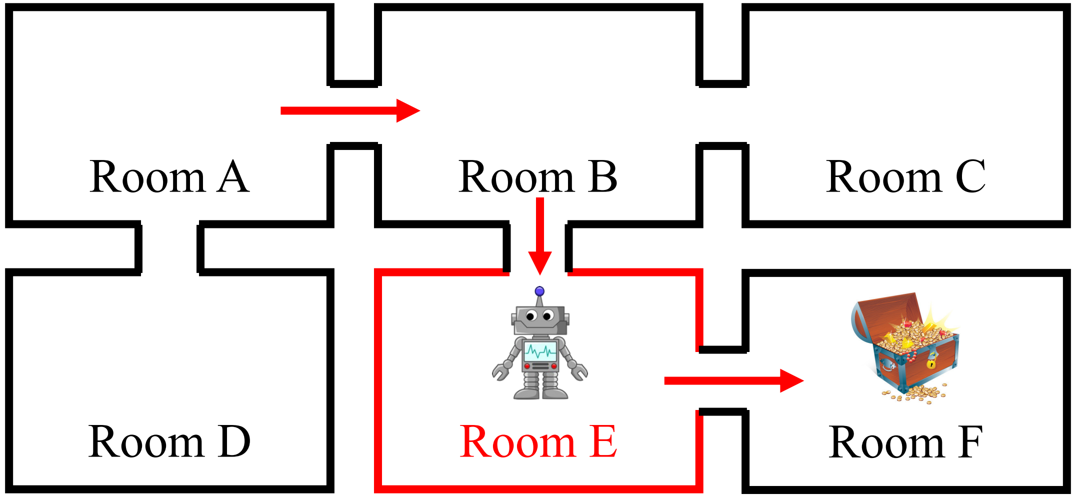
- **Policy** $\pi: [T] \times S \to A$
  - Maps room to direction to take

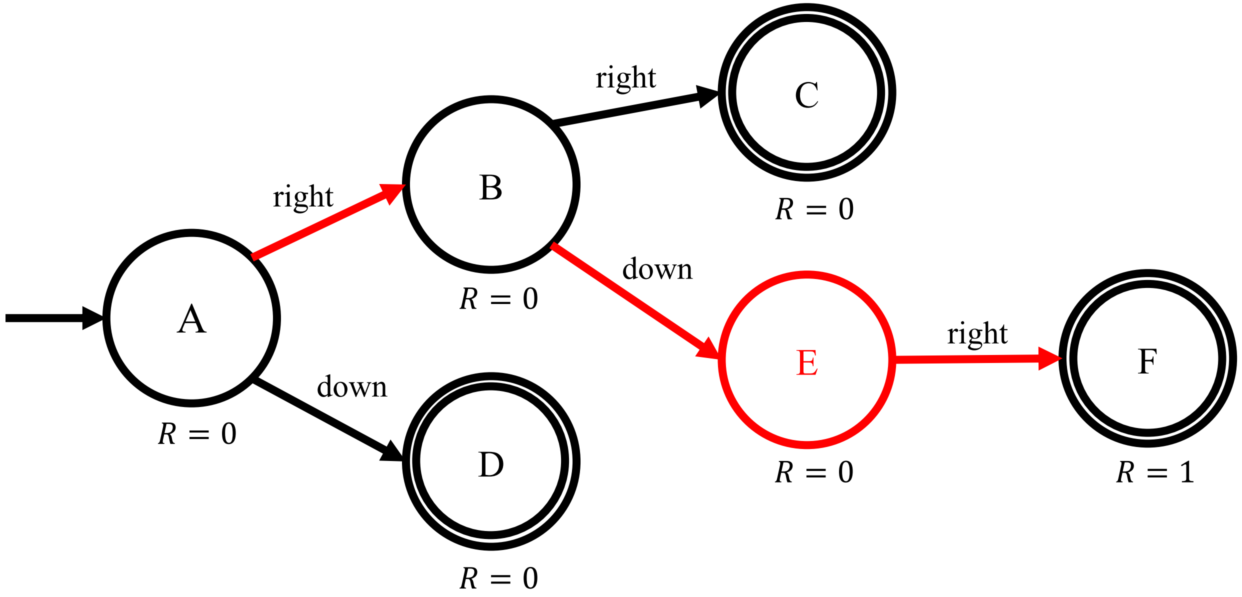




Markov Decision Processes (MDPs)

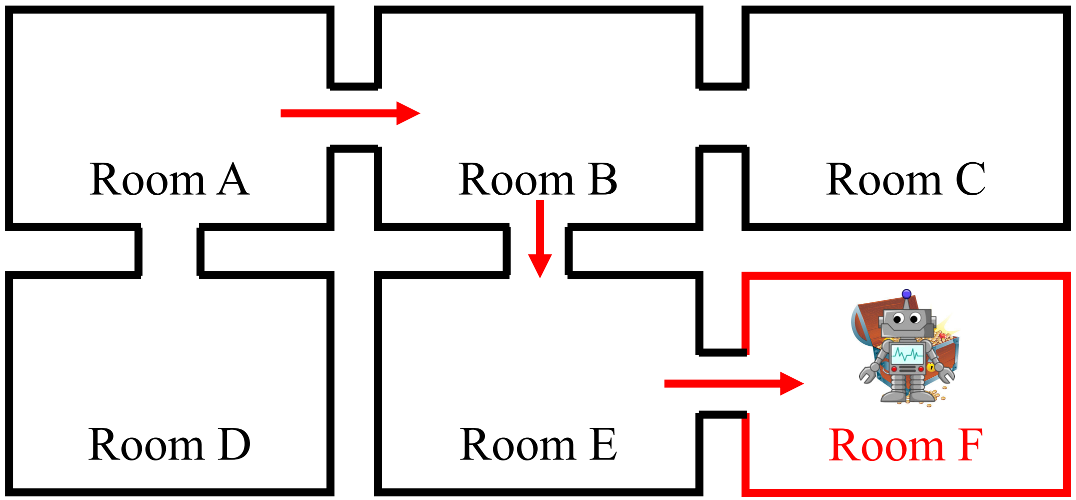
- **Policy** $\pi: [T] \times S \rightarrow A$
  - Maps room to direction to take

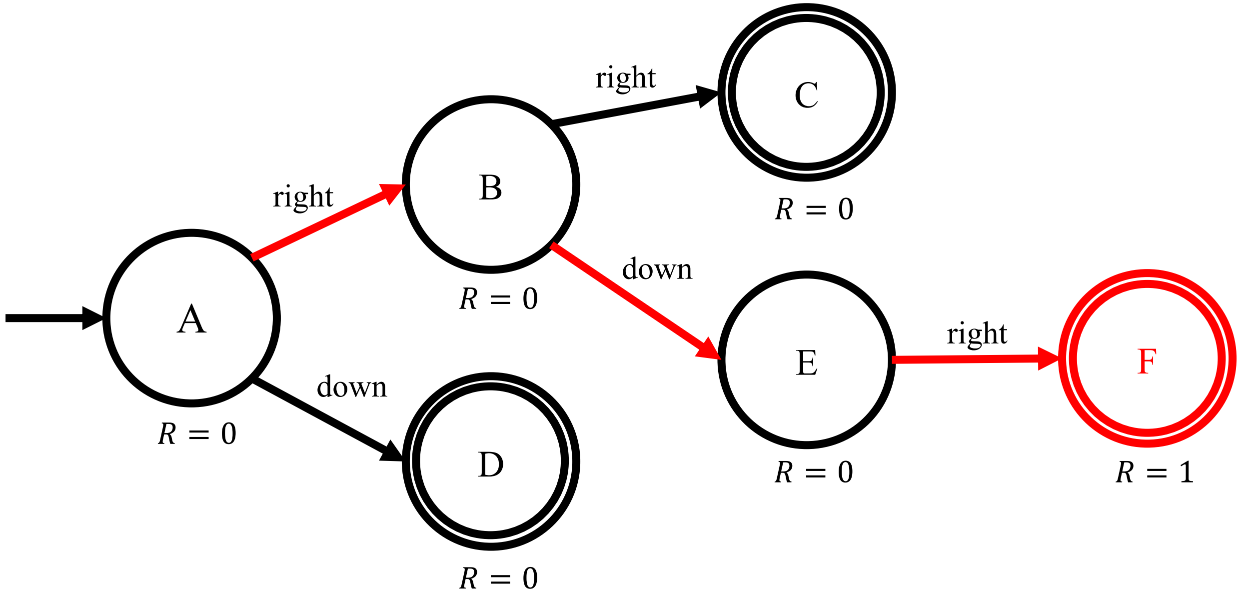




Markov Decision Processes (MDPs)

- **Policy** $\pi: [T]\times S \to A$
  - Maps room to direction to take





Reinforcement Learning

- **Goal:** Compute a policy π with high **cumulative reward**:

$$R^{(\pi)} = \sum_{t=1}^{T} R(s_t, a_t)$$

  - $s_1 = s_0$ is the initial state
  - $a_t = \pi(t, s_t)$ is the action taken
  - $s_{t+1} = P(s_t, a_t)$ is the state transition
  - T is the time horizon

- What about stochasticity?

Reinforcement Learning

- **Goal:** Compute a policy π with high **cumulative reward**:

$$R^{(\pi)} = \sum_{t=1}^{T} R(s_t, a_t)$$

  - $s_1 \sim P_0(\cdot)$ is the initial state
  - $a_t = \pi(t, s_t)$ is the action taken
  - $s_{t+1} = P(s_t, a_t)$ is the state transition
  - T is the time horizon

- What about stochasticity?

Reinforcement Learning

- **Goal:** Compute a policy π with high **cumulative reward**:

$$R^{(\pi)} = \sum_{t=1}^{T} R(s_t, a_t)$$

  - $s_1 \sim P_0(\cdot)$ is the initial state
  - $a_t \sim \pi(\cdot \mid t, s_t)$ is the action taken
  - $s_{t+1} = P(s_t, a_t)$ is the state transition
  - T is the time horizon

- What about stochasticity?

Reinforcement Learning

- **Goal:** Compute a policy π with high **cumulative reward**:

$$R^{(\pi)} = \sum_{t=1}^{T} R(s_t, a_t)$$

  - $s_1 \sim P_0(\cdot)$ is the initial state
  - $a_t \sim \pi(\cdot \mid t, s_t)$ is the action taken
  - $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ is the state transition
  - T is the time horizon

- What about stochasticity?

Reinforcement Learning

- **Goal:** Compute a policy π with high **cumulative reward**:

$$R^{(\pi)} = \mathbb{E}\left[\sum_{t=1}^{T} R(s_t, a_t)\right]$$

  - $s_1 \sim P_0(\cdot)$ is the initial state
  - $a_t \sim \pi(\cdot \mid t, s_t)$ is the action taken
  - $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ is the state transition
  - T is the time horizon

- What about stochasticity?

Fairness in Reinforcement Learning

- Note that $R^{(\pi)}$ is the reward for the decision maker (e.g., the bank), not for the individual

- We assume that there is additionally an **individual reward** $\rho(s, a)$, with corresponding **individual cumulative reward**

$$\rho^{(\pi)} = \mathbb{E}\left[\sum_{t=1}^{T} \rho(s_t, a_t)\right]$$

Fairness in Reinforcement Learning

- We assume each individual is associated with a subgroup $z \in \{0,1\}$
  - We assume z does not change over time
  - The reward for an individual in subgroup z is

$$\rho_z^{(\pi)} = \mathbb{E}\left[\sum_{t=1}^{T} \rho(s_t, a_t) \mid s_0 = (\tilde{s}_0, z)\right]$$

- π satisfies **demographic parity** if for all $z, z' \in \{0,1\}$, we have

$$\rho_z^{(\pi)} = \rho_{z'}^{(\pi)}$$

Imposing Fairness

- **Note:** Focus on planning, not learning
- Adding constraints to dynamic programming is hard!
- Typically, constrained MDPs are solved by formulating the cumulative reward objective as a **linear program**
- Then, we can impose fairness as a constraint in the linear programming

Linear Programming Formulation

- The **state-action** distribution of an MDP is

$$\lambda_{t,s,a}^{(\pi)} = \Pr[\text{ in state } s \wedge \text{take action } a \mid \text{on time step } t \ ]$$

- Defined recursively by

$$\lambda_{1,s,a}^{(\pi)} = P_0(s) \cdot \pi(\, a \mid 1, s\,)$$

$$\lambda_{t+1,s',a'}^{(\pi)} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a}^{(\pi)} \ \cdot P(\, s' \mid s, a\,) \cdot \pi(\, a' \mid t, s'\,)$$

Linear Programming Formulation

- **Idea:** Instead of computing π, compute the optimal $\lambda_{t,s,a}$
- Then, we have $\pi(\, a \mid t, s\,) = \frac{\lambda_{t,s,a}}{\sum_{a' \in A} \lambda_{t,s,a'}}$
- Given $\lambda_{t,s,a}$, the cumulative reward is

$$R = \sum_{t=1}^{T} \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot R(s, a)$$

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?

- For a given π, we have

$$\lambda_{1,s,a} = P_0(s) \cdot \pi(\, a \mid 1, s\,)$$

$$\lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(\, s' \mid s, a\,) \cdot \pi(\, a' \mid t, s'\,)$$

- Remember, π can be arbitrary here

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?

- For a given π, we have

$$\sum_{a \in A} \lambda_{1,s,a} = P_0(s) \cdot \sum_{a \in A} \pi(a \mid 1, s)$$

$$\lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s, a) \cdot \pi(a' \mid t, s')$$

- Remember, π can be arbitrary here

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?
- For a given π, we have

$$\sum_{a \in A} \lambda_{1,s,a} = P_0(s)$$

$$\lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s, a) \cdot \pi(a' \mid t, s')$$

- Remember, π can be arbitrary here

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?

- For a given π, we have

$$\sum_{a \in A} \lambda_{1,s,a} = P_0(s)$$

$$\lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s, a) \cdot \pi(a' \mid t, s')$$

- Remember, π can be arbitrary here

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?

- For a given π, we have

$$\sum_{a \in A} \lambda_{1,s,a} = P_0(s)$$

$$\sum_{a' \in A} \lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s, a) \cdot \sum_{a' \in A} \pi(a' \mid t, s')$$

- Remember, π can be arbitrary here

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?
- For a given π, we have

$$\sum_{a \in A} \lambda_{1,s,a} = P_0(s)$$

$$\sum_{a' \in A} \lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s, a)$$

- Remember, π can be arbitrary here

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?

- For a given π, we have

$$\sum_{a \in A} \lambda_{1,s,a} = P_0(s)$$

$$\sum_{a' \in A} \lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s, a)$$

- Remember, π can be arbitrary here

Linear Programming Formulation

- How to make sure λ represents an actual state-action distribution?

- For a given π, we have

$$\sum_{a \in A} \lambda_{1,s,a} = P_0(s)$$

$$\sum_{a' \in A} \lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s, a)$$

- Implicitly, also constrain $\lambda_{t,s,a} \in [0,1]$

Linear Programming Formulation

- Solve the following linear program:

$$
\begin{aligned}
&\arg\max_{\lambda} \textstyle\sum_{t=1}^{T} \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot R(s,a) \\
&\text{subj. to } \textstyle\sum_{a \in A} \lambda_{1,s,a} = P_0(s) \\
&\qquad\qquad \textstyle\sum_{a' \in A} \lambda_{t+1,s',a'} = \sum_{s \in S} \sum_{a \in A} \lambda_{t,s,a} \cdot P(s' \mid s,a)
\end{aligned}
$$

- **Note:** Dual of the more typical LP formulation

Imposing Fairness as a Constraint

- To impose fairness, add the constraint

$$p_z^{-1} \sum_{t=1}^{T} \sum_{(\tilde{s},z)\in S} \sum_{a\in A} \lambda_{t,s,a} \cdot \rho(s,a) = p_{z'}^{-1} \sum_{t=1}^{T} \sum_{(\tilde{s},z')\in S} \sum_{a\in A} \lambda_{t,s,a} \cdot \rho(s,a)$$

- Here, we have

$$p_z = \sum_{(\tilde{s},z)\in S} P_0\big((\tilde{s},z)\big)$$

Summary

- Algorithm for solving MDP by formulating it as a linear program
- Focus on computing the optimal state-action distribution
- Fairness is imposed as a constraint in the linear program

Agenda

- Selective compliance
- Fairness in sequential decision-making