6.S979 Topics in Deployable ML, Fall 2019

# Causal Inference and Predicting Counterfactuals II

David Sontag

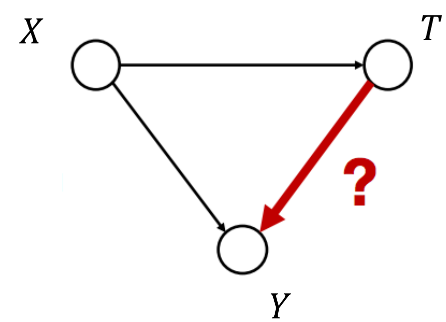
*Acknowledgement:* several slides adapted from Fredrik Johansson and Michael Oberst

CSAIL

imes
INSTITUTE FOR MEDICAL ENGINEERING & SCIENCE

# Reminder of 9/26 lecture: Causal effects

- ▸Potential outcomes under treatment and control, $Y(1), Y(0)$
- ▸Covariates and treatment, $X, T$




- ▸Conditional average treatment effect (CATE)

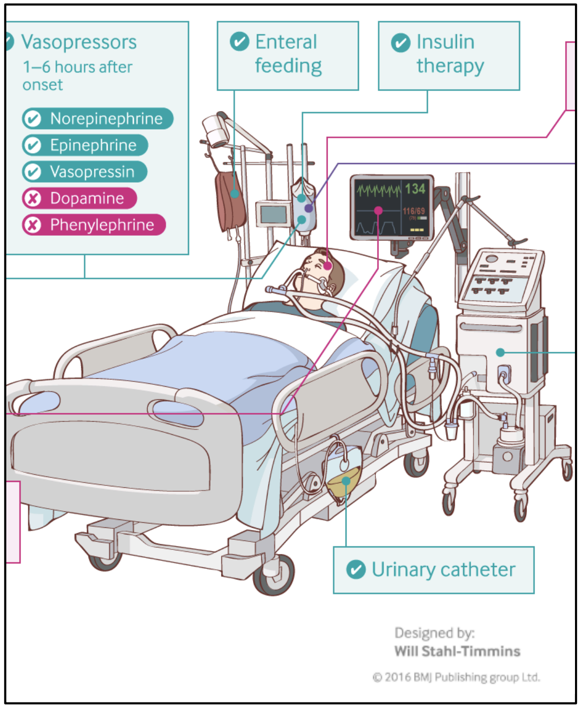
$$CATE(X) = \mathbb{E}[\underbrace{Y(1) - Y(0)}_{\text{Potential outcomes}} \mid \underbrace{X}_{\text{Features}}]$$

# Today: Sequential decision making

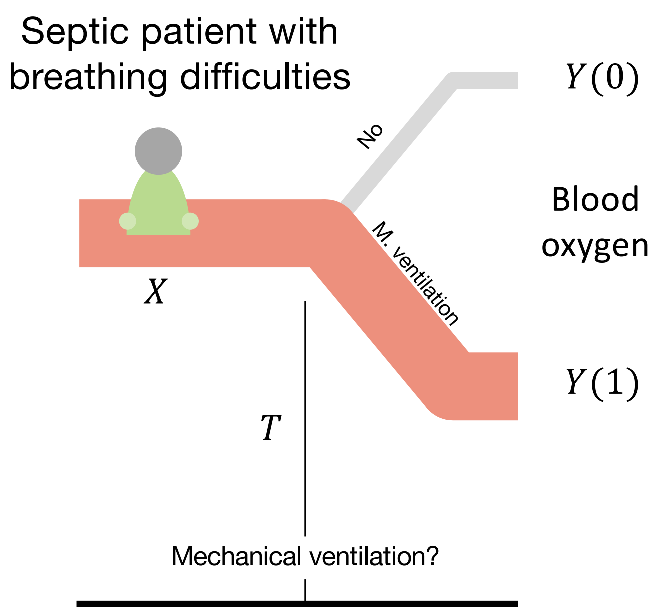
- A **policy $\pi$** assigns treatments to patients (typically depending on their medical history/state)
- **Single time-point example:**
  "Treat if effect is positive"
  For a patient with medical history $x$, $\pi(x) = \mathbb{I}[CATE(x) > 0]$
- Many clinical decisions are made in **sequence**
  - Choices early **may rule out** actions later
  - Can we optimize the **policy** by which actions are made?

# Example: Sepsis management

- **Sepsis** is a complication of an infection which can lead to massive organ failure and death
- One of the leading causes of death in the **ICU**
- Primary way to treat is to resolve the **infection, e.g. with antibiotics**
- Other symptoms need **management:** breathing difficulties, low blood pressure, …



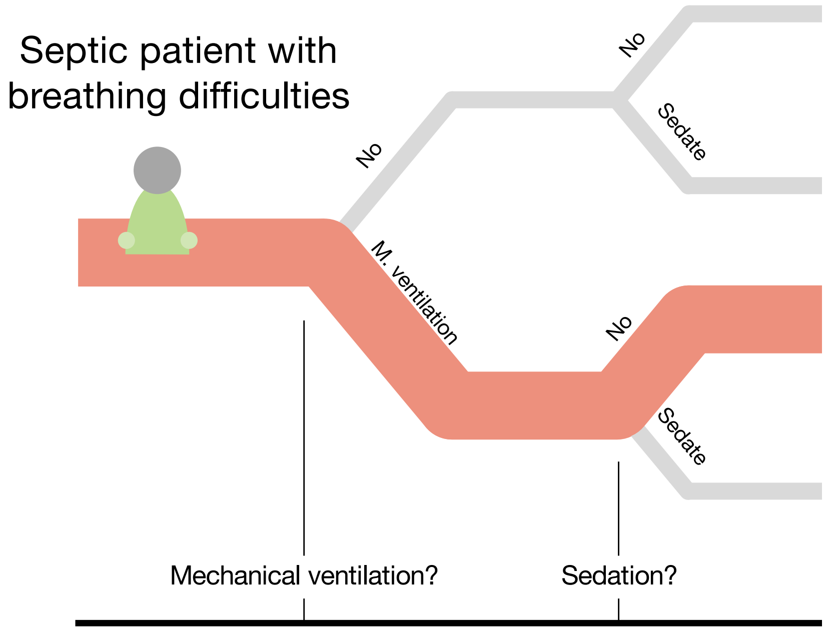
# Just one action? Easy!




1. Should the patient be put on mechanical ventilation?

*With a single action & outcome, suffices to directly reason about potential outcomes – reduce to what we know from 9/26 lecture*

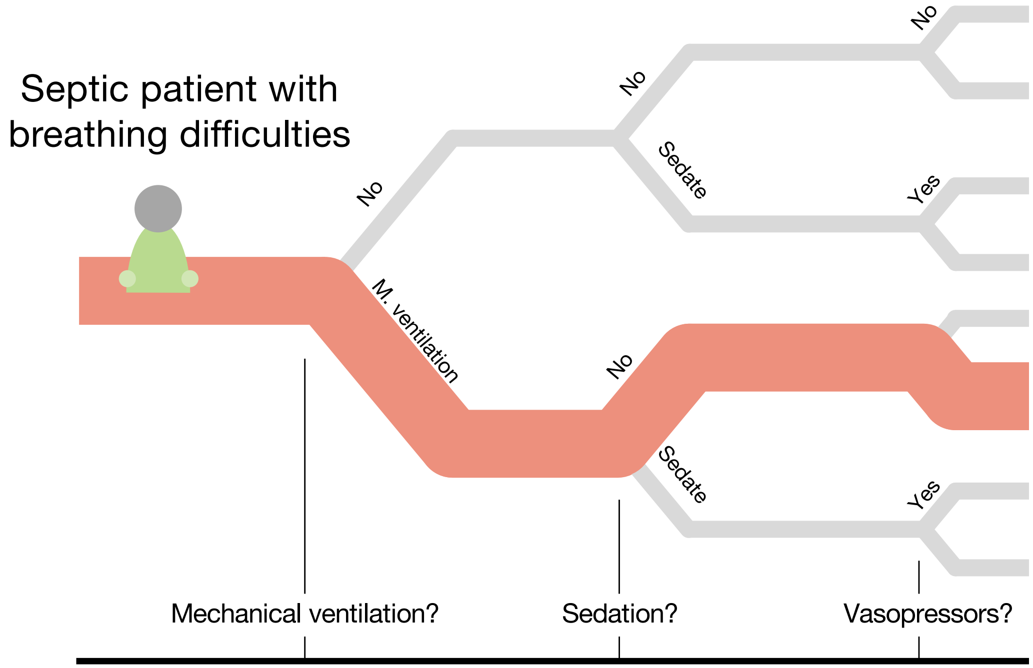
# Example: Sepsis management




2. Should the patient be sedated?

(To alleviate discomfort due to mech. ventilation)

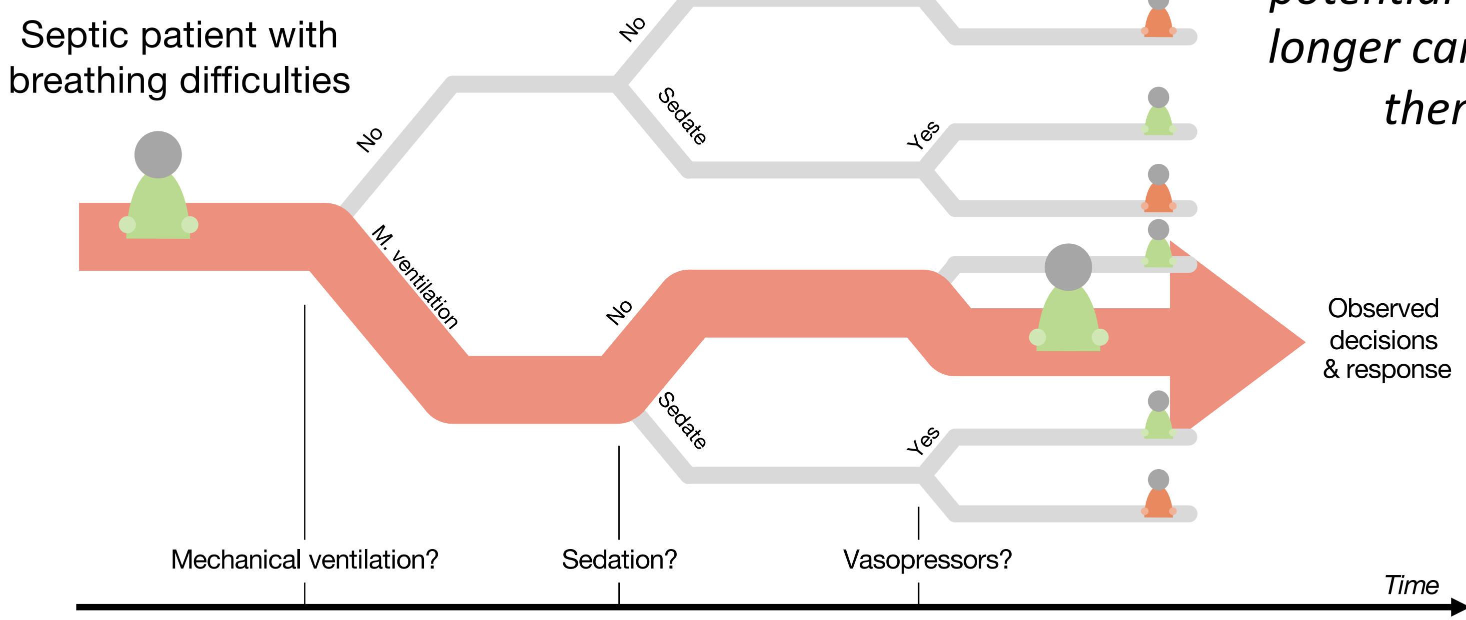
# Example: Sepsis management




3. Should we artificially raise blood pressure?

(Which may have dropped due to sedation)

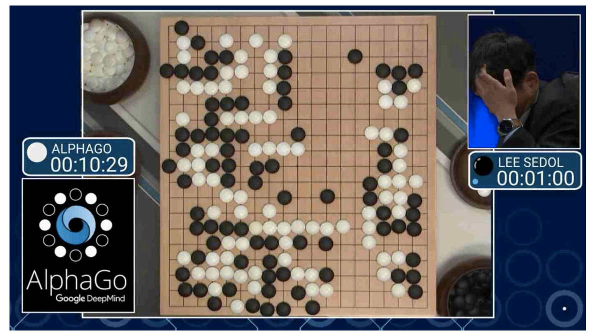
# Example: Sepsis management




*Exponentially many potential outcomes – no longer can reason about them directly*

# No prob, we’ll use reinforcement learning

- ▸AlphaStar
- ▸AlphaGo
- ▸DQN Atari
- ▸Open AI Five




# Reinforcement learning

- Maximize reward!

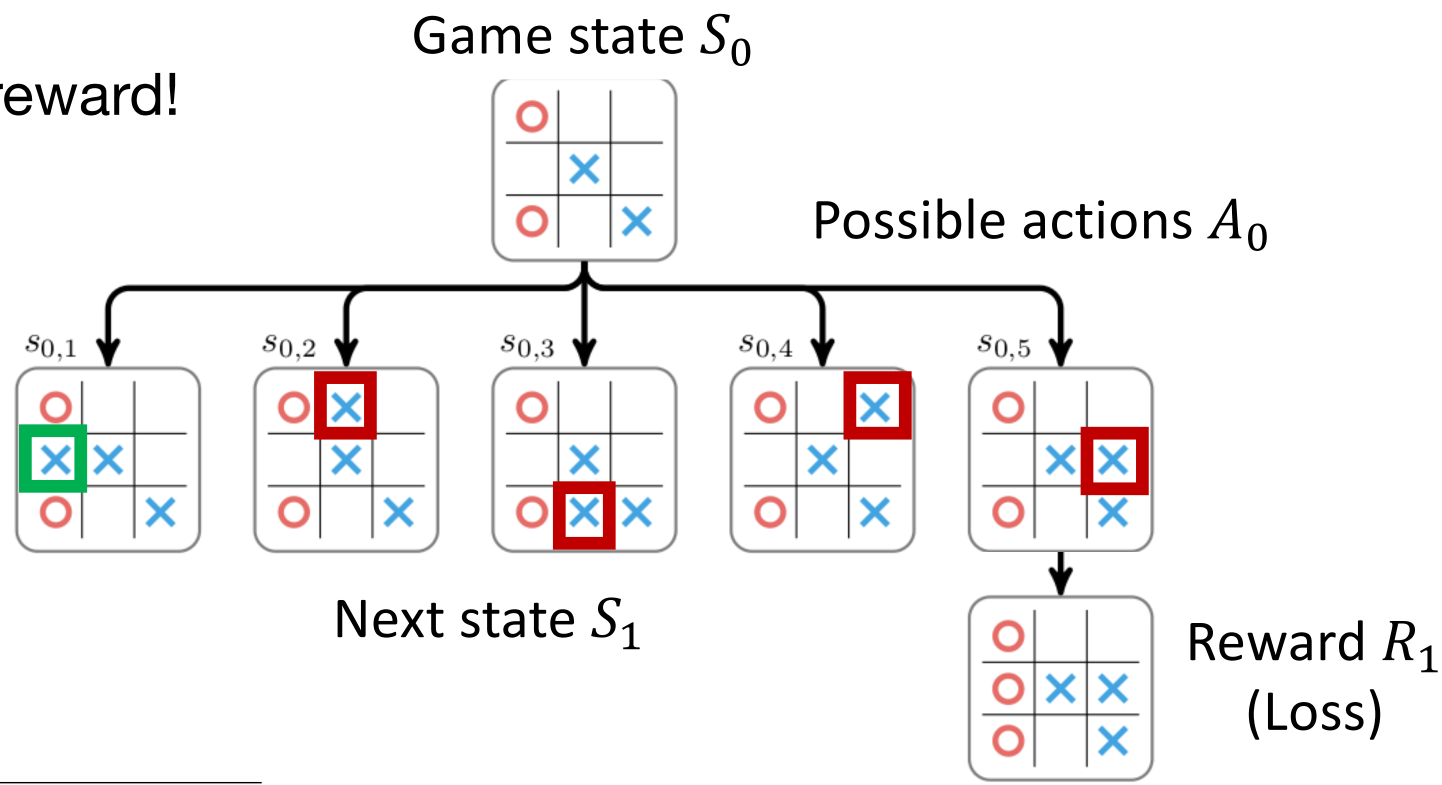
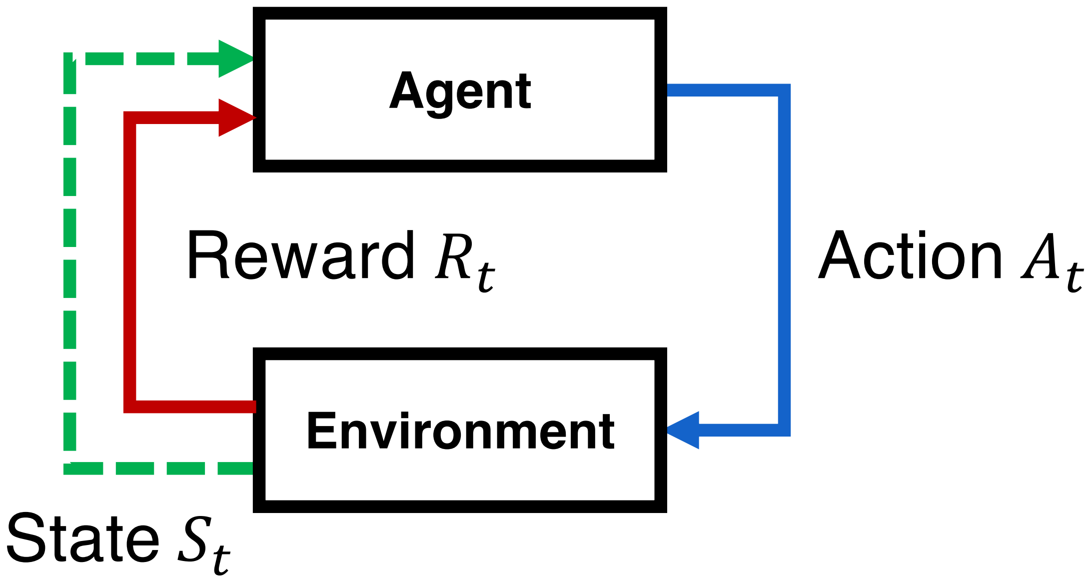


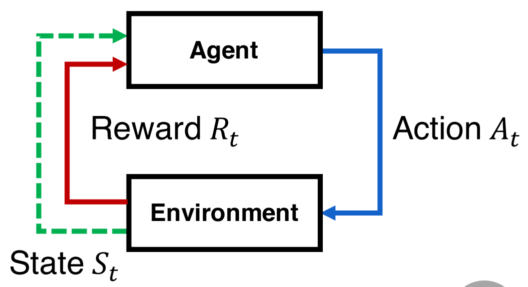

Figure by Tim Wheeler, tim.hibal.org

# Decision processes

- An **agent** repeatedly, at times $t$ takes **actions** $A_t$ to receive **rewards** $R_t$ from an **environment**, the **state** $S_t$ of which is (partially) observed



# Decision process: Mechanical ventilation




$$R_t = R_t^{vitals} + R_t^{vent\,off} + R_t^{vent\,on}$$

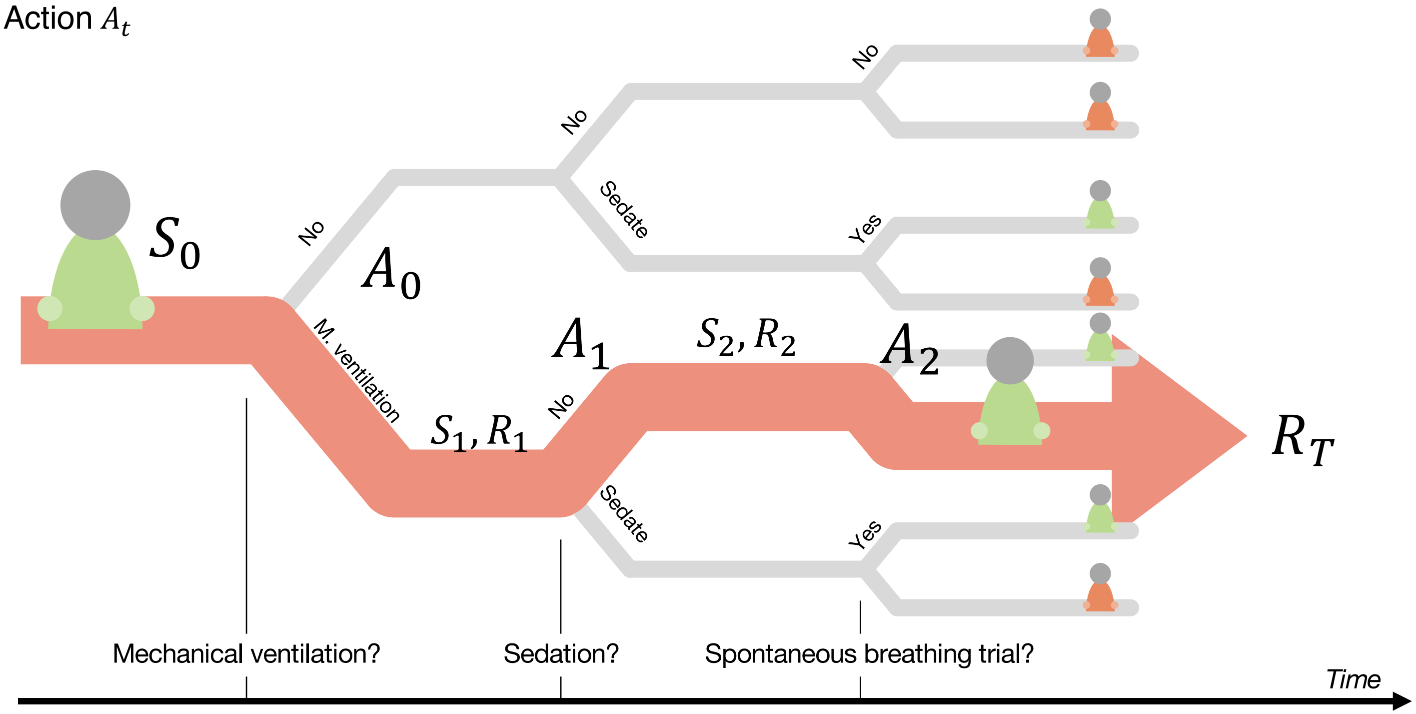

A Reinforcement Learning Approach to Weaning of Mechanical Ventilation in Intensive Care Units

Niranjani Prasad, Princeton University; Li-Fang Cheng, Princeton University; Corey Chivers, Penn Medicine; Michael Draugelis, Penn Medicine; Barbara E. Engelhardt, Princeton University

arXiv:1704.06300v1 [cs.AI] 20 Apr 2017

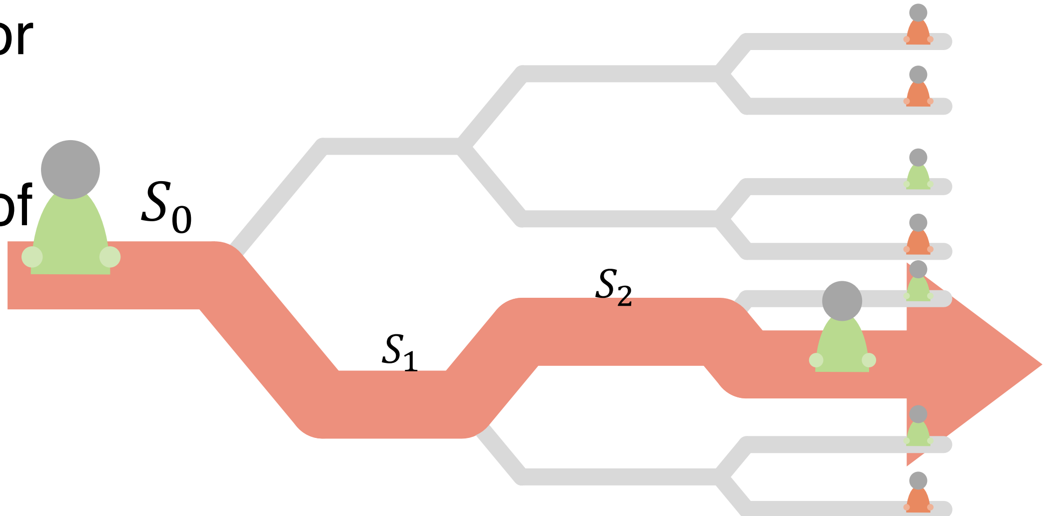
Abstract

1 Introduction



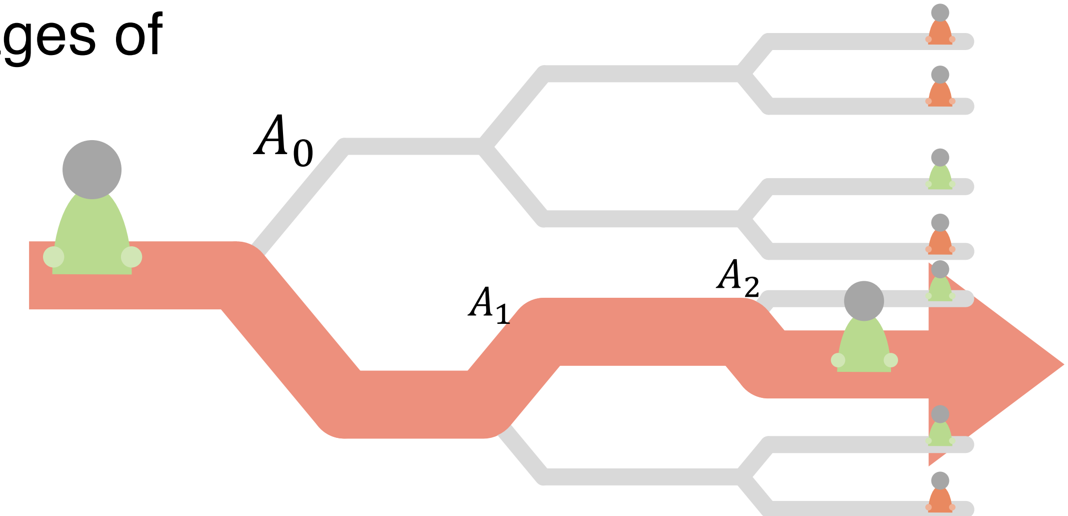
# Decision process: Mechanical ventilation

- **State** $S_t$ includes demographics, physiological measurements, ventilator settings, level of consciousness, dosage of sedatives, time to ventilation, number of intubations



# Decision process: Mechanical ventilation

- **Actions** $A_t$ include intubation and extubation, as well as administration and dosages of sedatives



# Decision processes

- A decision process specifies how states $S_t$, actions $A_t$, and rewards $R_t$ are **distributed**: $p(S_0, \dots, S_T, A_0, \dots, A_T, R_0, \dots, R_T)$

- The agent interacts with the environment according to a **behavior policy** $\mu = p(A_t \mid \cdots)^*$

* The … depends on the type of agent

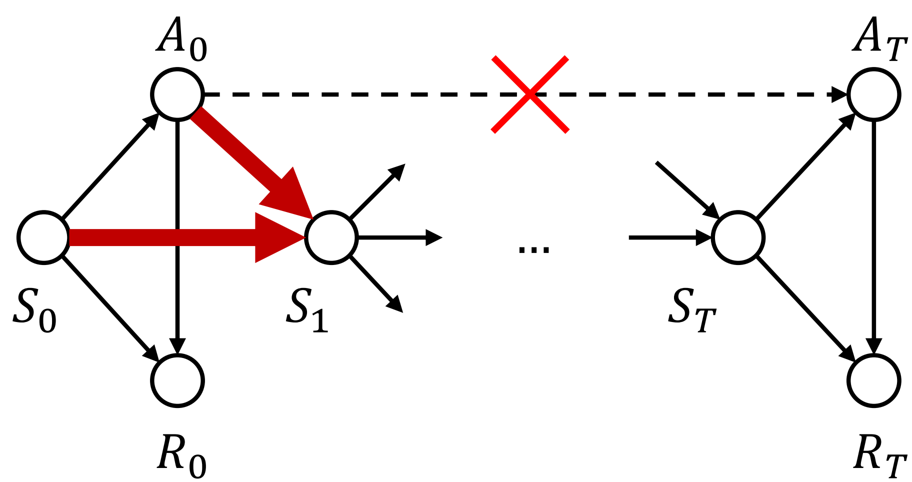
# Markov Decision Processes

- ▸Markov decision processes (MDPs) are a special case

- ▸Markov **transitions**: $p(S_t \mid S_0, \dots, S_{t-1}, A_0, \dots, A_{t-1}) = p(S_t \mid S_{t-1}, A_{t-1})$
- ▸Markov **reward** function: $p(R_t \mid S_0, \dots, S_{t-1}, A_0, \dots, A_{t-1}) = p(R_t \mid S_{t-1}, A_{t-1})$
- ▸Markov **action** policy $\mu = p(A_t \mid S_0, \dots, S_t, A_0, \dots, A_{t-1}) = p(A_t \mid S_t)$

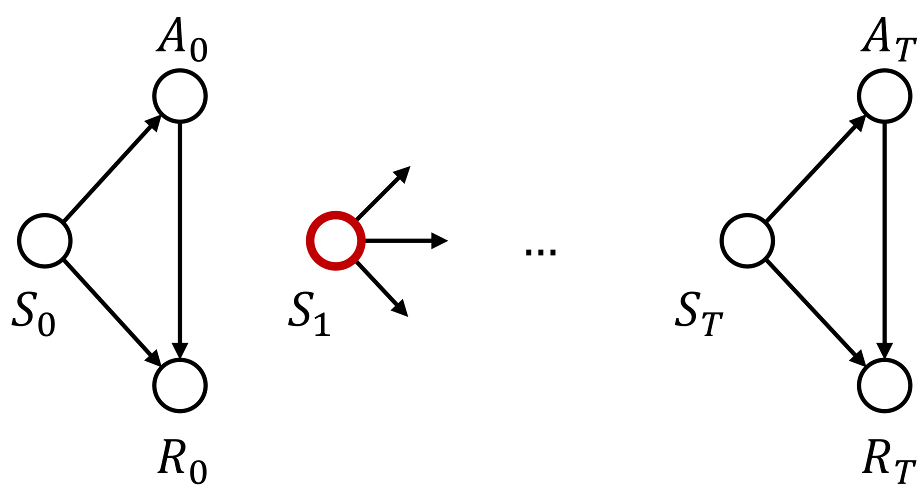
# Markov assumption

- State transitions, actions and reward depend only on most recent state-action pair



# Contextual bandits (special case)*

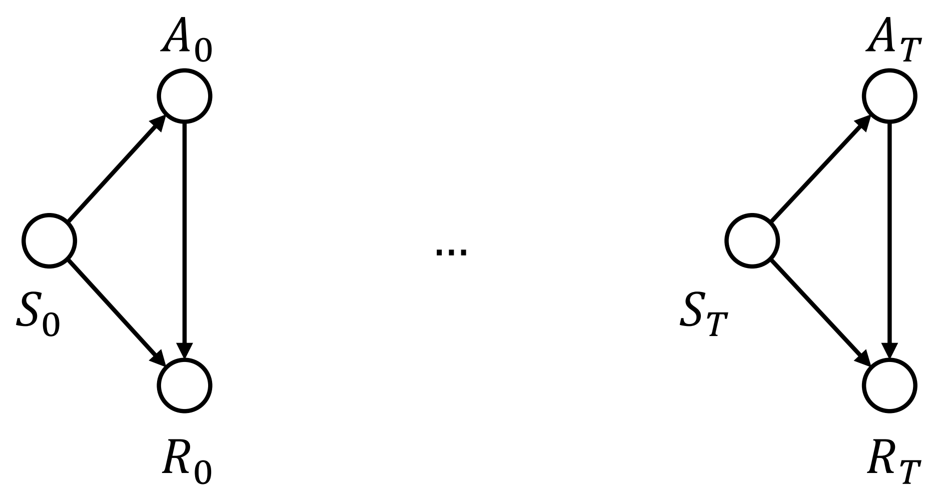
- ▶States are independent: $p( S_t \mid S_{t-1}, A_{t-1} ) = p(S_t)$
- ▶Equivalent to **single-step case**: potential outcomes!




* The term "contextual bandits" has connotations of efficient exploration, which is not addressed here

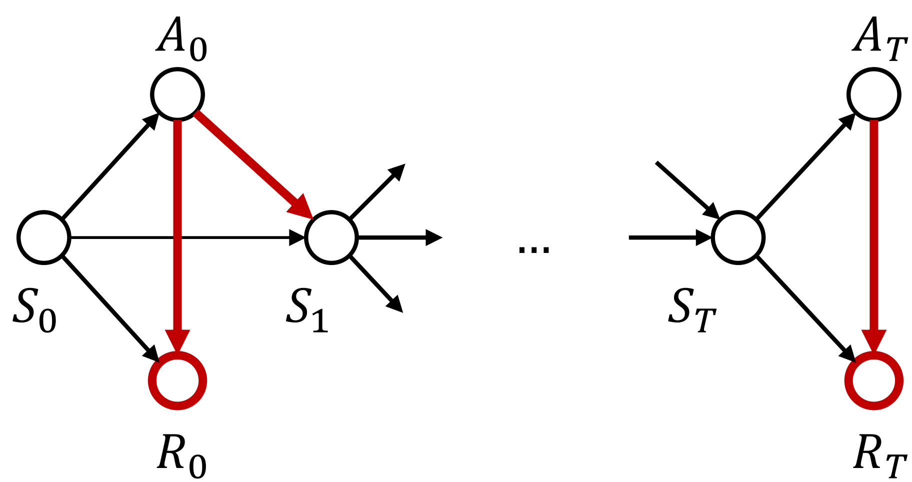
# Contextual bandits & potential outcomes

► Think of each state $S_i$ as an i.i.d. patient, the actions $A_i$ as the treatment group indicators and $R_i$ as the outcomes



# Goal of RL

- Like previously with causal effect estimation, we are interested in the effects of actions $A_t$ on future rewards



# Maximize expected cumulative reward

- The goal of most RL algorithms is to maximize the expected cumulative reward—the **value** $V_\pi$ of its policy $\pi$

- **Return**: $G_t = \sum_{s=t}^{T} R_s$

- **Value:** $V_\pi = \mathbb{E}_{A_t \sim \pi}[G_0]$

Sum of future rewards

Expected sum of rewards under policy $\pi$

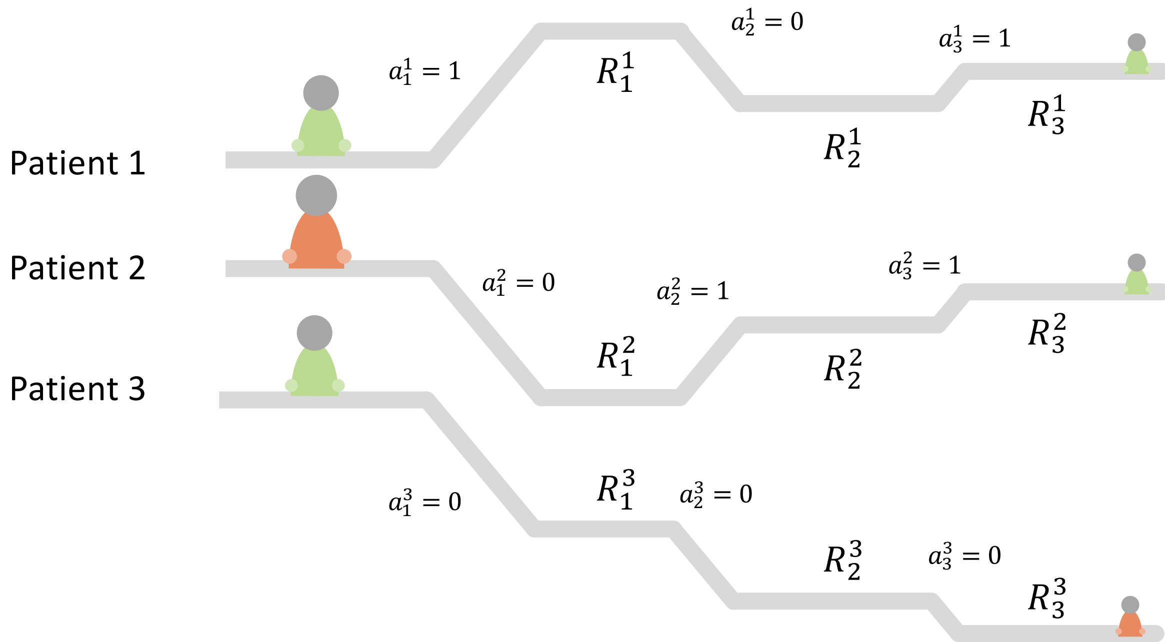
- The expectation is taken with respect to scenarios acted out according to the learned **policy** $\pi$

# Example

- Let's say that we have data from a policy $\mu$

Value

$$V_\mu \approx \frac{1}{n}\sum_{i=1}^{n} G^n$$




Return

$$G^1 = R_1^1 + R_2^1 + R_3^1$$

$$G^2 = R_1^2 + R_2^2 + R_3^2$$

$$G^3 = R_1^3 + R_2^3 + R_3^3$$

1. Decision processes
2. **Reinforcement learning**
3. Learning from batch (off-policy) data
4. Reinforcement learning in healthcare

# Paradigms*

**Model-based RL**

Transitions

$p(S_t \mid S_{t-1}, A_{t-1})$

G-computation

MDP estimation

**Value-based RL**

Value/return

$p(G_t \mid S_t, A_t)$

**Q-learning**

G-estimation

**Policy-based RL**

Policy

$p(A_t \mid S_t)$

REINFORCE

Marginal structural models

*We focus on off-policy RL here

# Paradigms*

| Model-based RL | Value-based RL | Policy-based RL |
|---|---|---|
| Transitions $p(S_t \mid S_{t-1}, A_{t-1})$ | Value/return $p(G_t \mid S_t, A_t)$ | Policy $p(A_t \mid S_t)$ |
| G-computation<br>MDP estimation | **Q-learning**<br>G-estimation | REINFORCE<br>Marginal structural models |

*We focus on off-policy RL here

# Q-learning

- Q-learning is a value-based reinforcement learning method
- The value of a **state-action pair** $(s, a)$ is

$$Q_\pi(s, a) \coloneqq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$$

(the expectation is over future states and rewards, for future actions taken according to $\pi$)

---

*Mathematical tool more than anything

# Q-learning

- Instead of directly optimizing over $\pi$, Q-learning optimizes over functions $Q(s, a)$. $\pi$ is assumed to be the deterministic policy

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

- The best $Q$ is the best **state-action value** function

$$Q^*(s, a) =: \max_{\pi} Q_{\pi}(s, a)$$

# Bellman equation

▸For the optimal Q-function $Q^*$, "**Bellman optimality**" holds

$$Q^*(s,a) = \mathbb{E}_\pi\left[R_t + \gamma \max_{a'} Q^*(S_{t+1}, a') \mid S_t = s, A_t = a\right]$$

State-action value     Immediate reward     Future (discounted) rewards*

▸Look for functions with this property!

# Q-learning (from last Thursday, 10/10)

---

**Algorithm 3** Q-learning

---

$Q_0(s,a) \leftarrow 0$ for all $s \in S, a \in A$
**for** $k = 1 \dots N$ **do**
Collect sample $(s, a, s', \hat{r})$ by playing with a policy induced from $Q_k$ (we will discuss choices for this policy)
$\hat{Q}(s,a) \leftarrow \hat{r} + \gamma \max_{a' \in A} Q(s', a')$
$Q_{k+1}(s,a) \leftarrow (1-\alpha) Q_k(s,a) + \alpha \hat{Q}(s,a)$
**end for**

---

- Fitted Q-learning
  - If $s$ is not discrete, we cannot maintain a table for $Q(s, a)$
  - Instead, we may represent $Q(s, a)$ by a **function** $Q_\theta$

# Q-learning (from last Thursday, 10/10)

---

**Algorithm 3** Q-learning

---

$Q_0(s,a) \leftarrow 0$ for all $s \in S, a \in A$
**for** $k = 1 \dots N$ **do**
Collect sample $(s, a, s', \hat{r})$ by playing with a policy induced from $Q_k$ (we will discuss choices for this policy)
$\hat{Q}(s,a) \leftarrow \hat{r} \cancel{+ \gamma \max_{a' \in A} Q(s', a')}$
$Q_{k+1}(s,a) \leftarrow (1-\alpha) Q_k(s,a) + \alpha \hat{Q}(s,a)$
**end for**

---

**If only single time/action, fitted Q-learning is identical to covariate adjustment**

- Fitted Q-learning
  - If $s$ is not discrete, we cannot maintain a table for $Q(s,a)$
  - Instead, we may represent $Q(s,a)$ by a **function** $Q_\theta$

1. Decision processes
2. Reinforcement learning paradigms
3. **Learning from batch (off-policy) data**
4. Reinforcement learning in healthcare

# Off-policy learning

- Trajectories $(s_1, a_1, r_1), \ldots, (s_T, a_T, r_T)$,of states $s_t$, actions $a_t$, and rewards $r_t$ observed in e.g. medical record

- Actions are drawn according to a behavior policy $\mu$, but we want to know the value of a new policy $\pi$

- Learning policies from this data is **at least as hard** as estimating treatment effects from observational data

# Assumptions for (off-policy) RL

▸Sufficient conditions for identifying value function

**Single-step case**

**Strong ignorability**:

$Y(0), Y(1) \perp\!\!\!\perp T \mid X$

"No *hidden* confounders"

**Overlap**:

$\forall x, t \colon p(T = t \mid X = x) > 0$

"All actions possible"

**Sequential case**

**Sequential randomization**:

$G(\ldots) \perp\!\!\!\perp A_t \mid \bar{S}_t, \bar{A}_{t-1}$

"Reward indep. of policy given history"

**Positivity**:

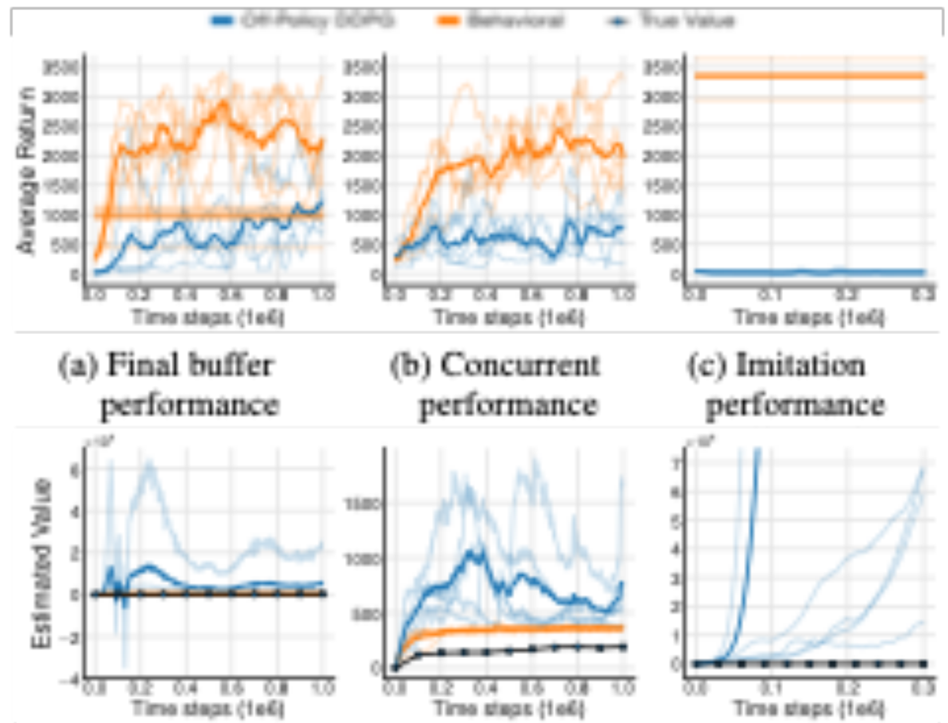
$\forall a, t \colon p(A_t = a \mid \bar{S}_t, \bar{A}_{t-1}) > 0$

"All actions possible at all times"

# The problem of overlap shows up all over deep RL

"Our results demonstrate that the performance of a state of the art deep actor-critic algorithm, DDPG (Lillicrap et al., 2015), deteriorates rapidly when the data is uncorrelated... These results suggest that off-policy deep reinforcement learning algorithms are ineffective when learning truly off-policy."

Fujimoto, Meger, Precup, ICML 2019



(a) Final buffer performance (b) Concurrent performance (c) Imitation performance

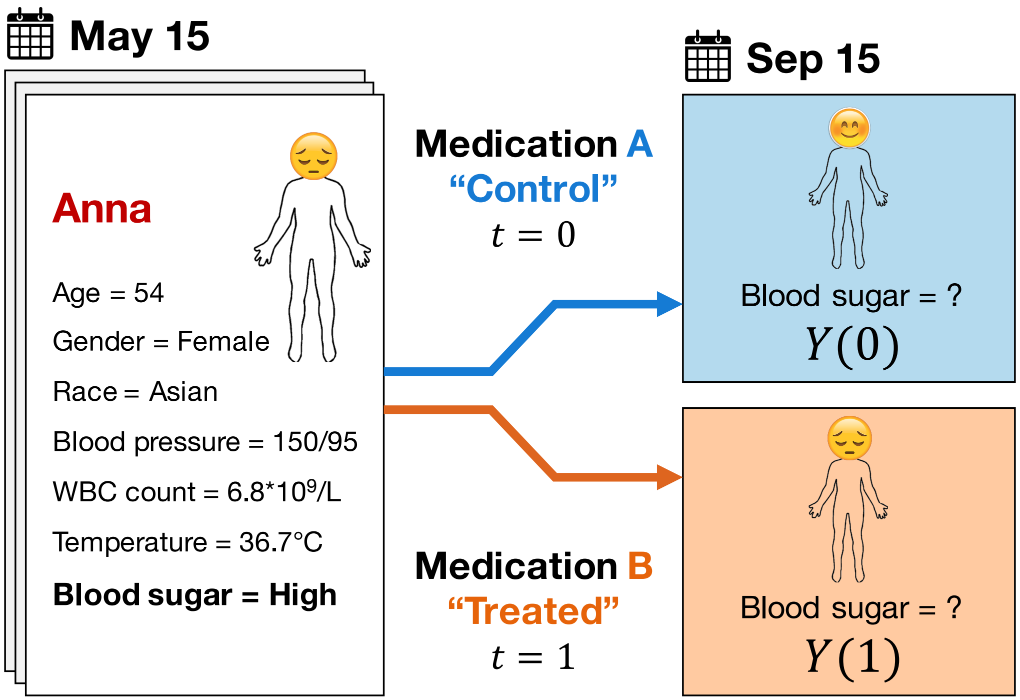
(d) Final buffer value estimate (e) Concurrent value estimate (f) Imitation value estimate

# Assumptions for (off-policy) RL

- Sufficient conditions for identifying value function

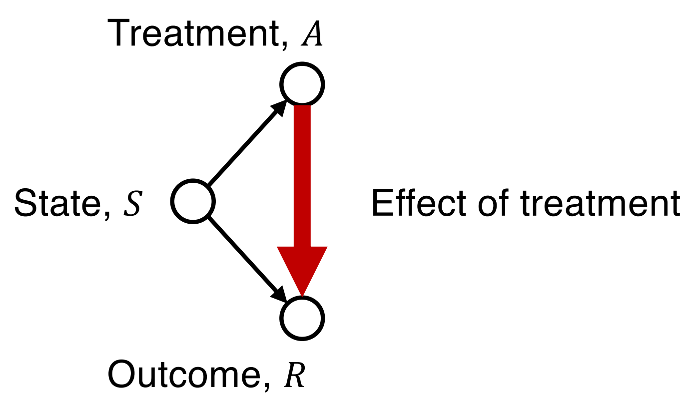
| Single-step case | Sequential case |
| --- | --- |
| **Strong ignorability**: | **Sequential randomization**: |
| $Y(0), Y(1) \perp\!\!\!\perp T \mid X$ | $G(\ldots) \perp\!\!\!\perp A_t \mid \bar{S}_t, \bar{A}_{t-1}$ |
| "No *hidden* confounders" | "Reward indep. of policy given history" |
| **Overlap**: | **Positivity**: |
| $\forall x, t\colon\ p(T = t \mid X = x) > 0$ | $\forall a, t\colon\ p(A_t = a \mid \bar{S}_t, \bar{A}_{t-1}) > 0$ |
| "All actions possible" | "All actions possible at all times" |

# Recap: Learning potential outcomes



# Treating Anna once

- We assumed a simple causal graph. This let us identify the causal effect of treatment on outcome from observational data



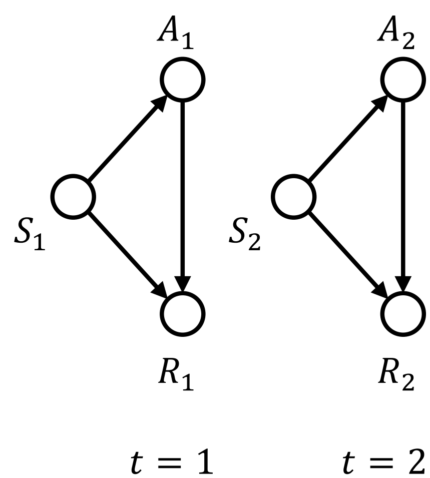

**Ignorability**

$$R(a) \perp\!\!\!\perp A \mid S$$

Potential outcome under action $a$

# Treating Anna over time

- Let's add a time point…



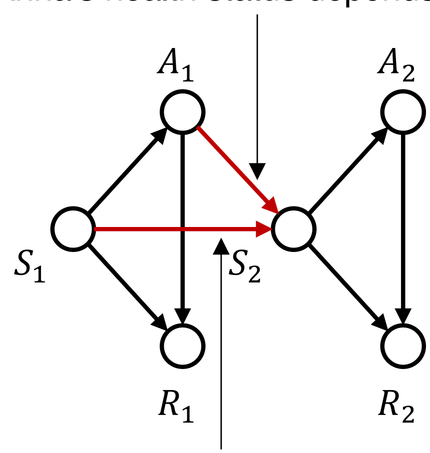

**Ignorability**

$R_t(a) \perp\!\!\!\perp A_t \mid S_t$

# Treating Anna over time

- What influences her state?

Anna's health status depends on how we treated her




It is likely that if Anna is diabetic, she will remain so

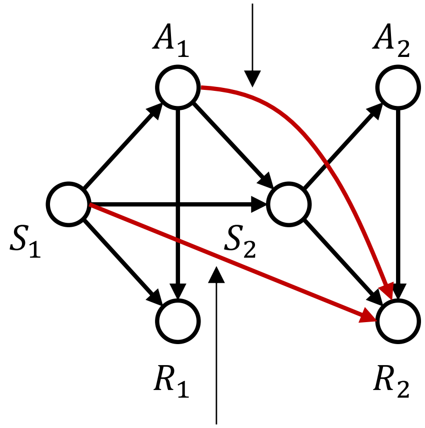
**Ignorability**

$R_t(a) \perp\!\!\!\perp A_t \mid S_t$

# Treating Anna over time

- What influences her state?

The outcome at a later time point may depend on earlier choices




The outcome at a later time may depend on an earlier state

**~~Ignorability~~**

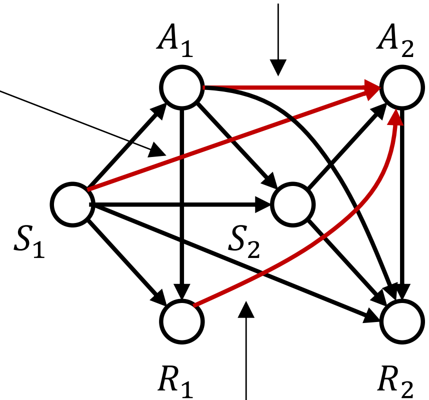
$\cancel{R_t(a) \perp\!\!\!\perp A_t \mid S_t}$

# Treating Anna over time

- What influences her state?

If we know that a patient had a symptom previously, it may affect future decisions

If we already tried a treatment, we might not try it again



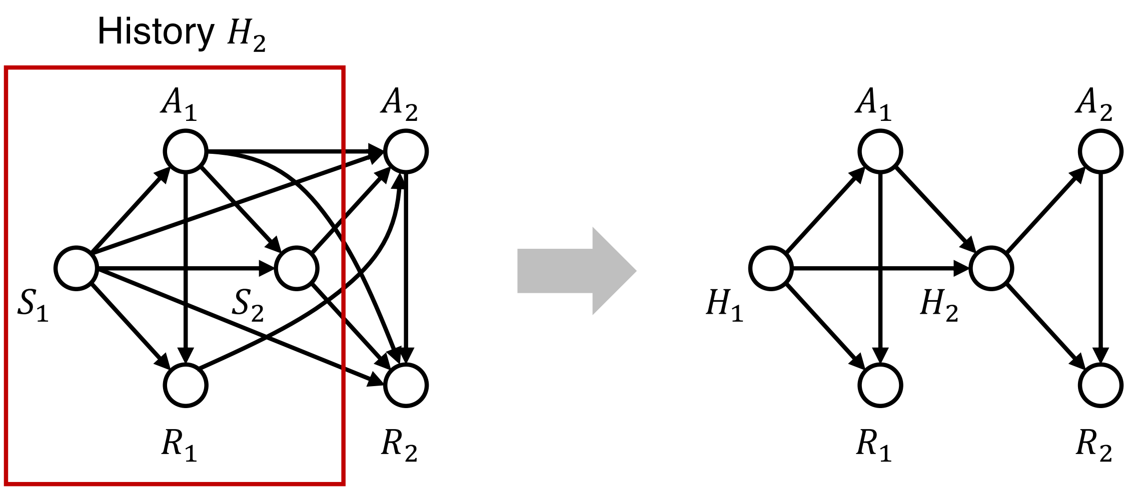

If the last treatment was unsuccessful, it may change our next choice

~~**Ignorability**~~

$\cancel{R_t(a) \perp\!\!\!\perp A_t \mid S_t}$

# State & ignorability

- To have sequential ignorability, we need to remember history!




**Ignorability**

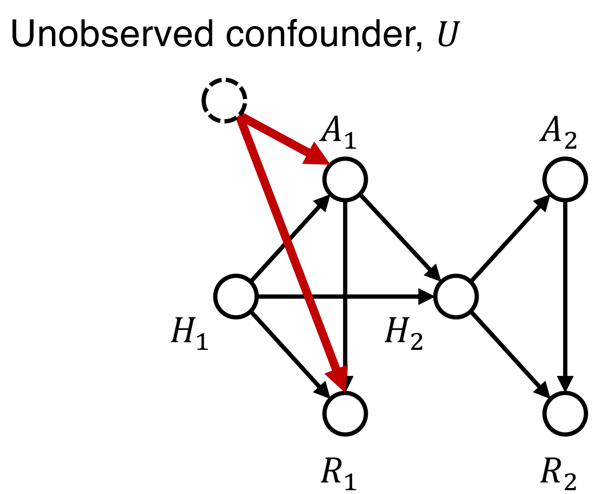
$R_t(a) \perp\!\!\!\perp A_t \mid H_t$

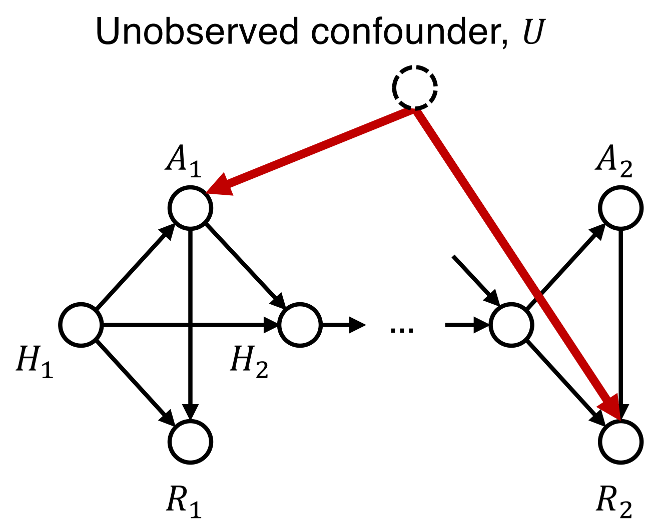
# Summarizing history

- The difficulty with history is that its **size grows with time**
- A simple change of the standard MDP is to store the states and actions of a **length $k$ window** looking backwards
- Another alternative is to **learn a summary** function that maintains what is relevant for making optimal decisions, e.g., using an RNN

# State & ignorability

- We cannot leave out unobserved confounders



# What made success possible/easier?

▸ **Full observability**
*Everything important to optimal action is observed*

▸ **Markov** dynamics
*History is unimportant given recent state(s)*

▸ Limitless **exploration** & self-play through simulation
*We can test "any" policy and observe the outcome*

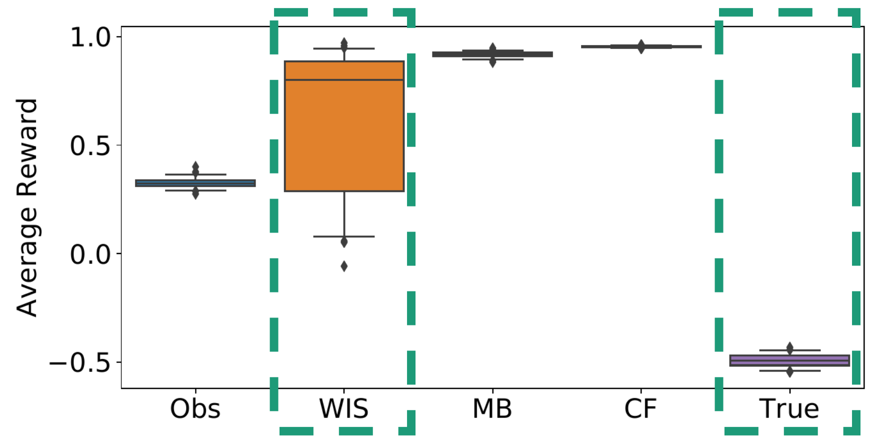
▸ **Noise-less** state/outcome (for games, specifically)

# How do we build trust in RL policies?

- **Goal**: Apply reinforcement learning in high risk settings (e.g., healthcare)
- **Problem**: How to safely evaluate a policy? No simulator, and off-policy evaluation can fail due to
  - Unobserved confounding
  - Small sample sizes & lack of overlap
  - Poorly specified rewards

# Building trust in RL policies

- **Goal**: Apply reinforcement learning in high risk settings (e.g., healthcare)
- **Problem**: How to safely evaluate a policy?  No simulator, and off-policy evaluation can fail due to
  - Unobserved confounding
  - Small sample sizes & lack of overlap
  - Poorly specified rewards
- Could try to interpret the policy directly, but if not possible, what can we do?
- **Approach: look at the proposed policy in the context of a specific individual**

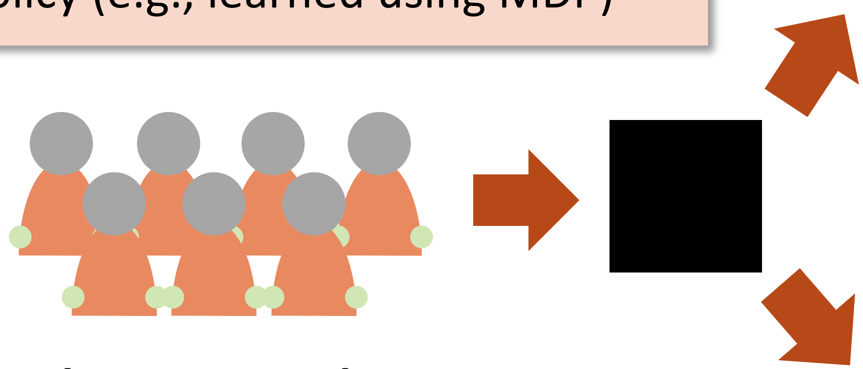



**Obs:** Observed Reward of behavior policy
**WIS:** Weighted Importance Sampling
**MB**: Model-Based Rollouts
**CF**: Counterfactual Rollouts
**True**: Actual RL reward, not known

# Building trust in RL policies

Suppose we are given:

- Markov Decision Process (MDP)
- Policy (e.g., learned using MDP)



**Observational Data**

**Markov Decision Process (MDP)**

$P(S', R \mid S, A)$

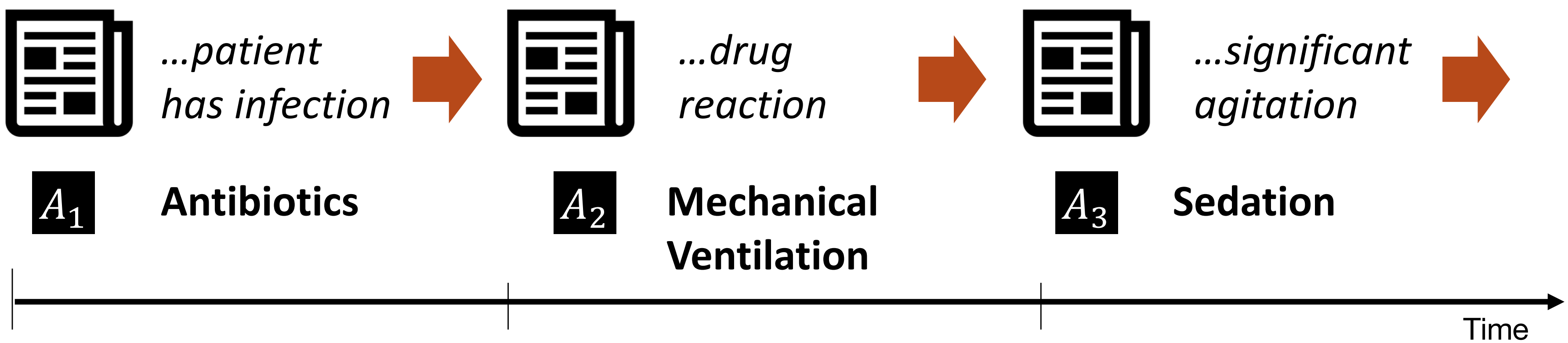
$S$: Current State
$A$: Action
$R$: Reward
$S'$: Next State

**Policy**

$\pi(A \mid S)$

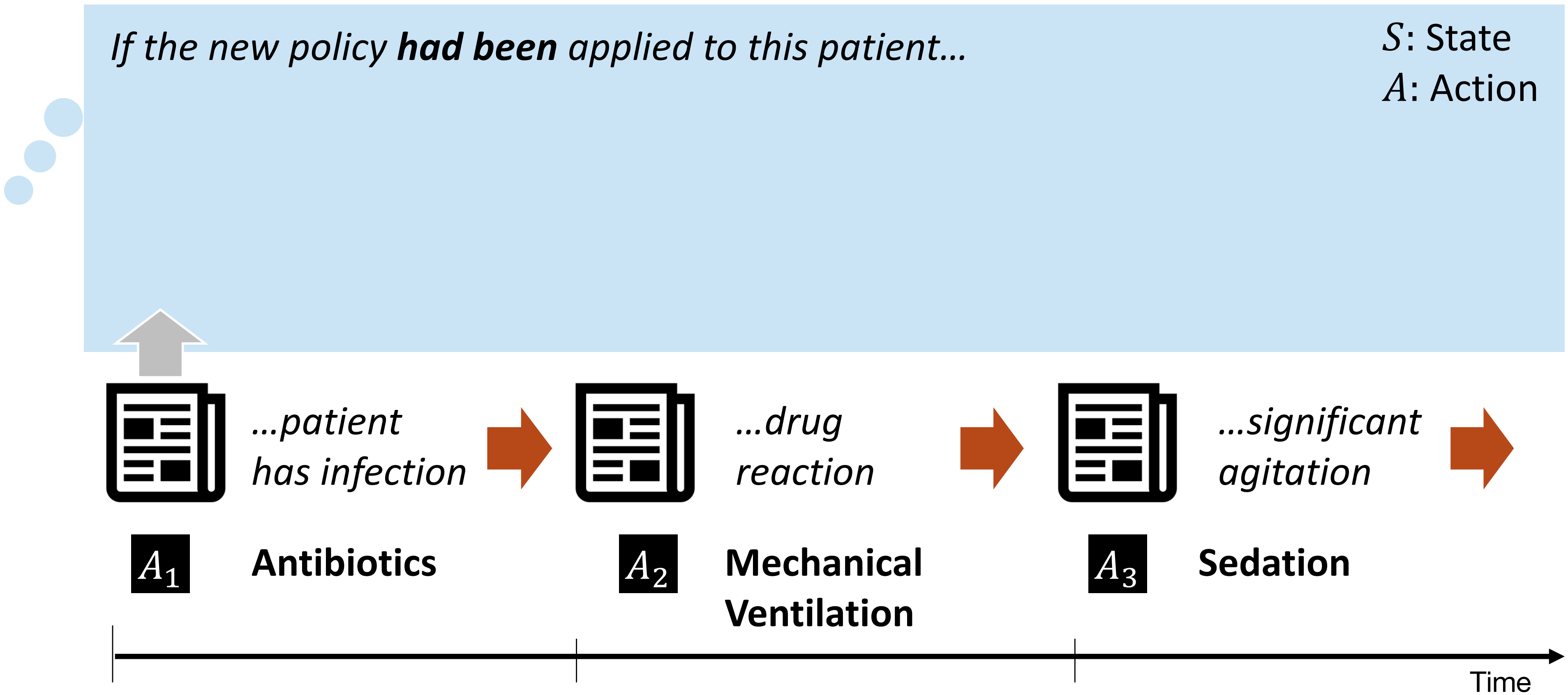
# Using counterfactuals to “sanity check”

$S$: State
$A$: Action



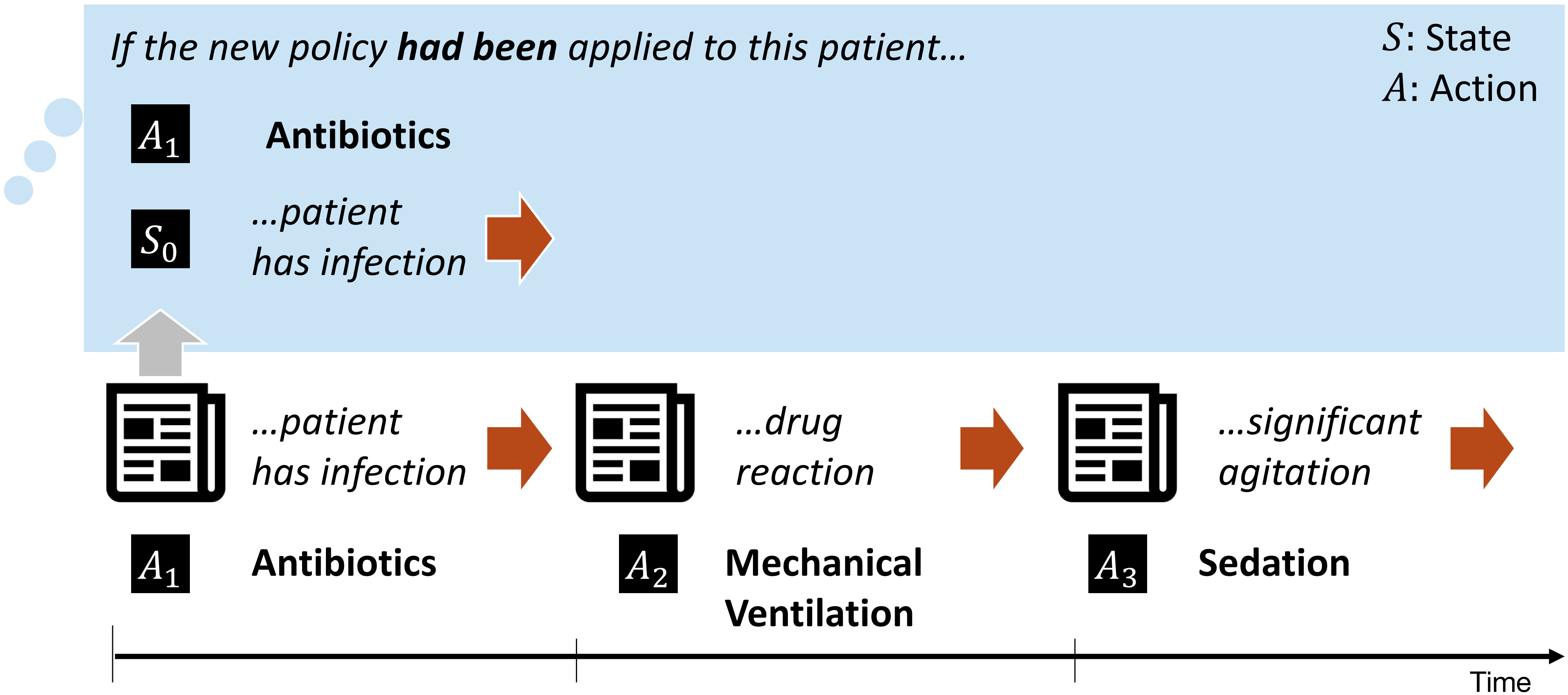

[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”



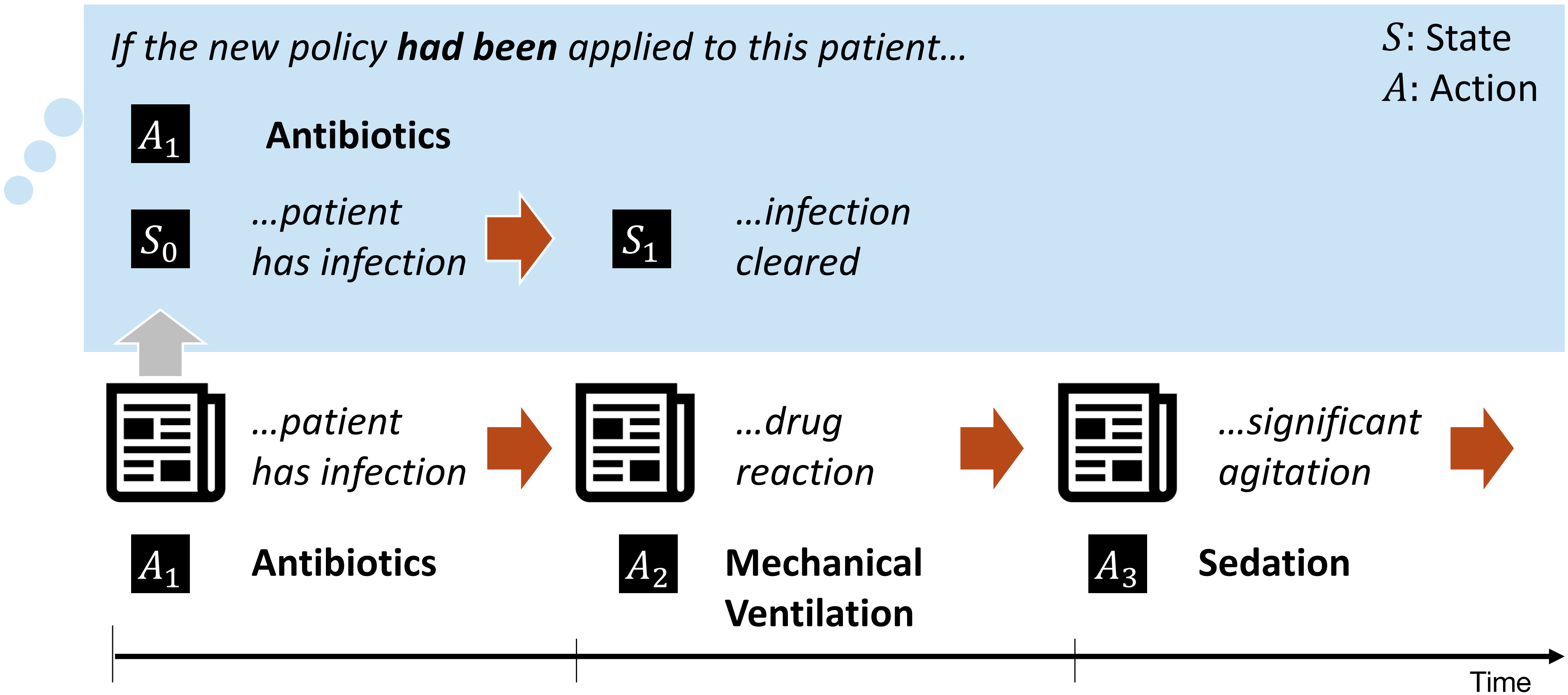

[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”



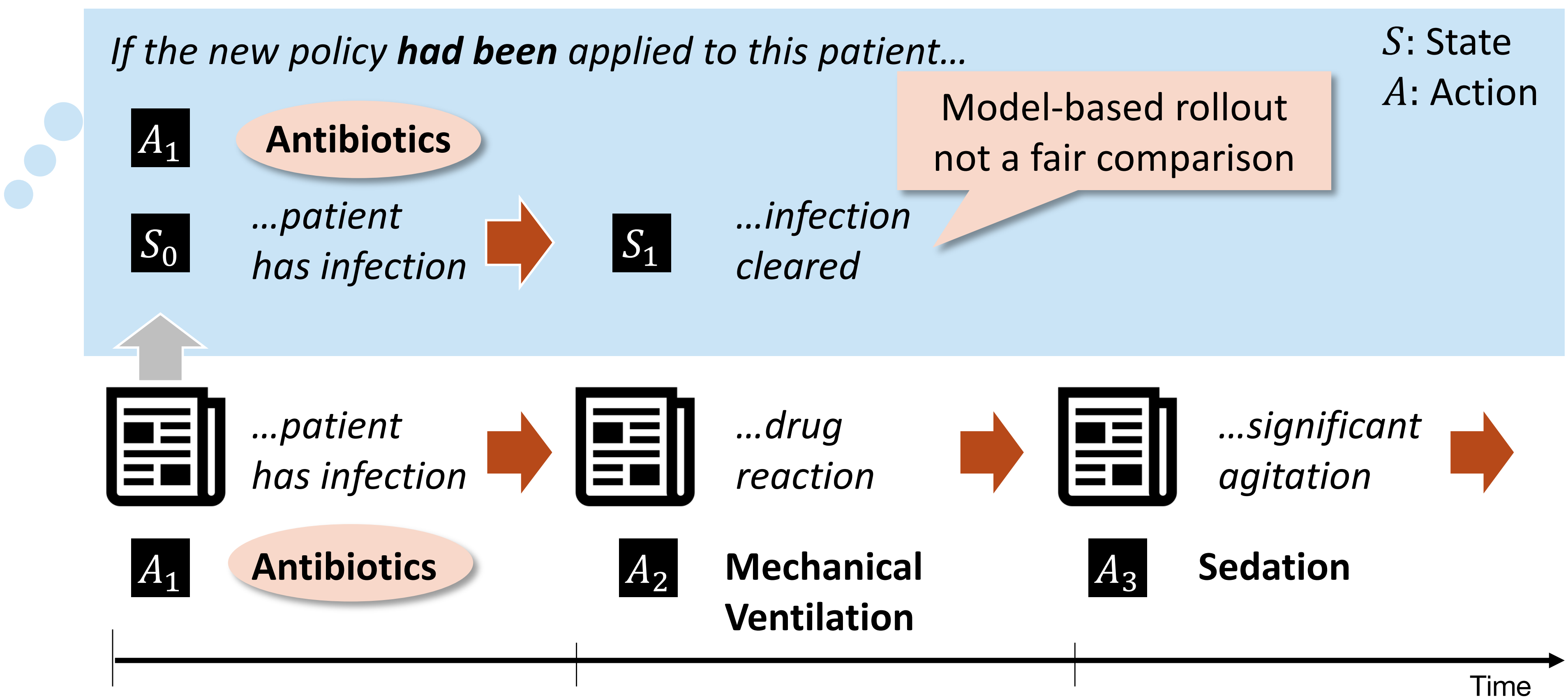

[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”



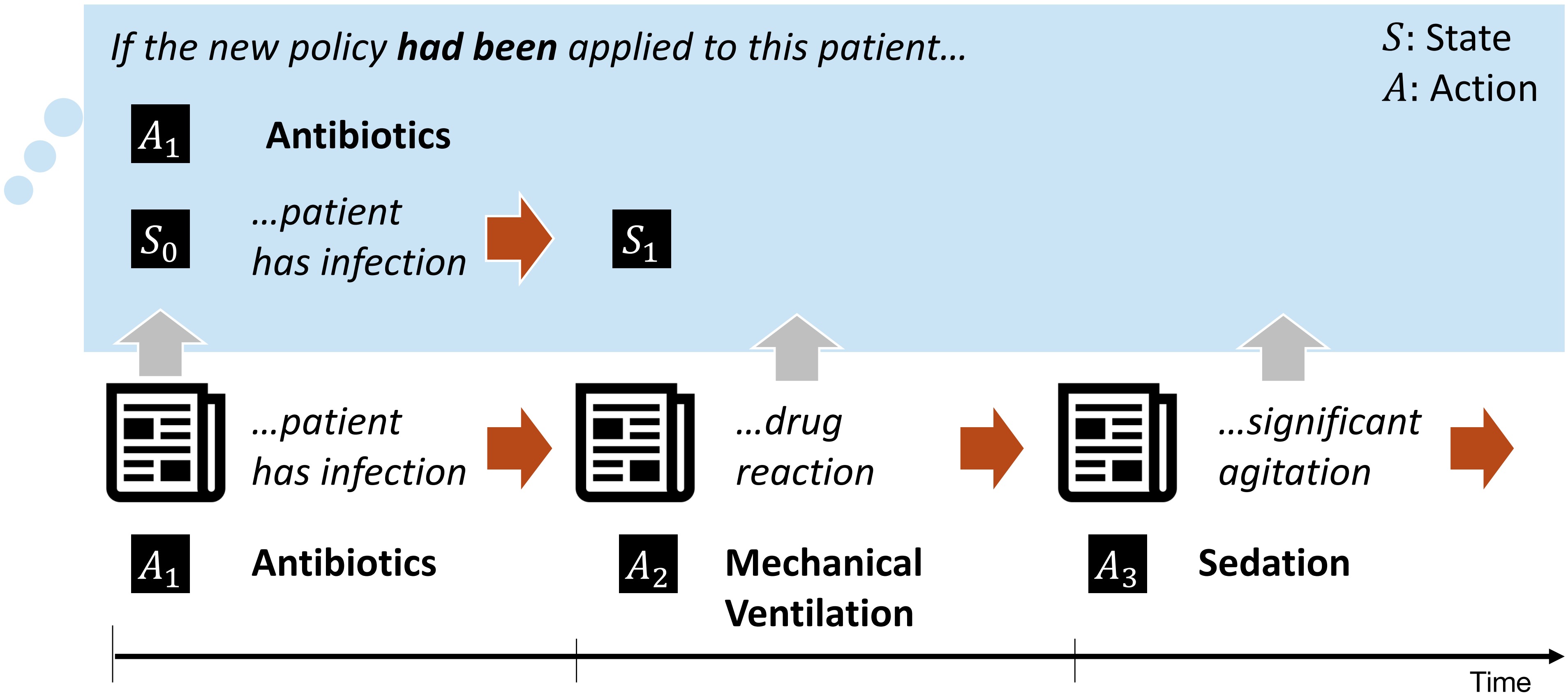

[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”



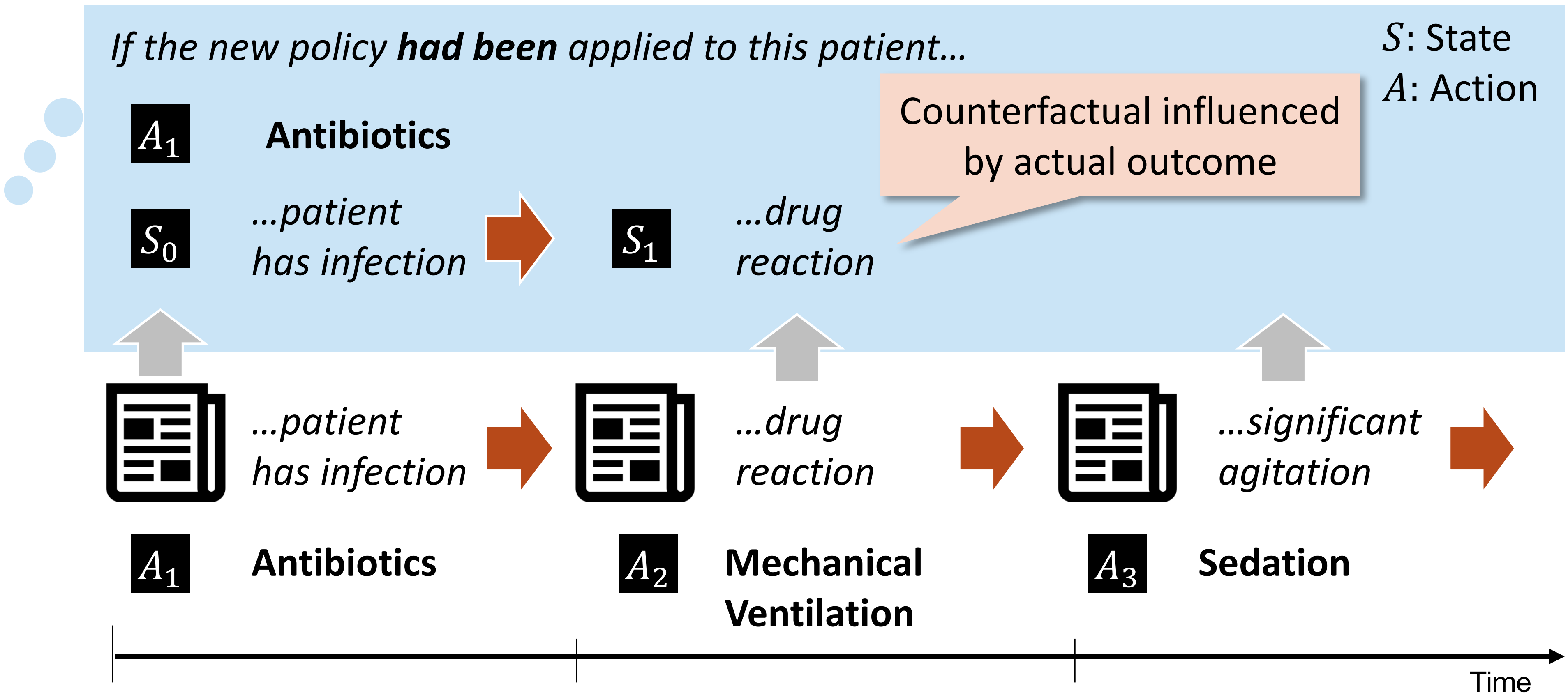

[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”



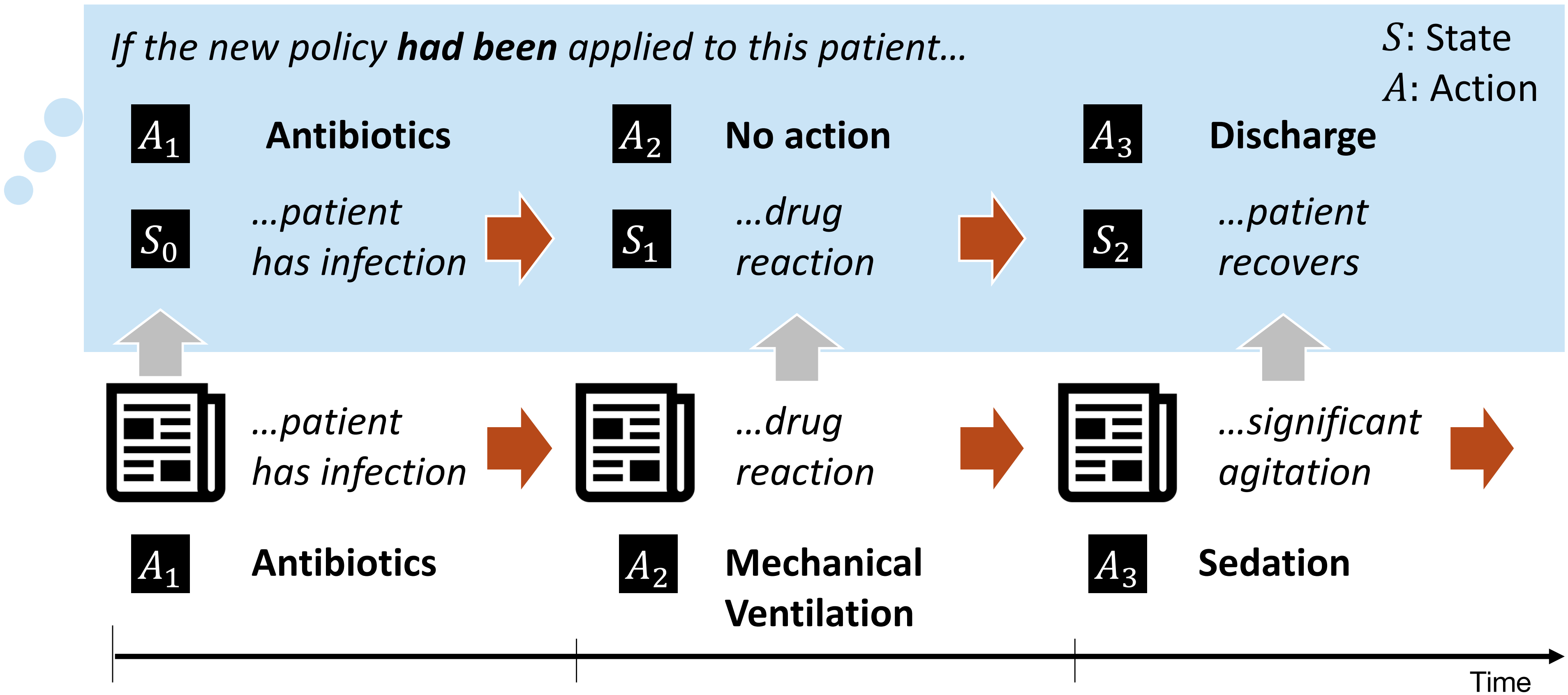

[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”




[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”




[Balke & Pearl, 1994]

# Using counterfactuals to “sanity check”

*If the new policy **had been** applied to this patient…*

$S$: State
$A$: Action

$A_1$ **Antibiotics** | $A_2$ **No action** | $A_3$ **Discharge**

$S_0$ *…patient has infection* | $S_1$ *…drug reaction* | $S_2$ *…patient recovers*

*…patient has infection* | *…drug reaction* | *…significant agitation*

$A_1$ **Antibiotics** | $A_2$ **Mechanical Ventilation** | $A_3$ **Sedation**

**Idea:** If the counterfactual trajectory is unreasonable given full context of patient, the model / policy may be flawed

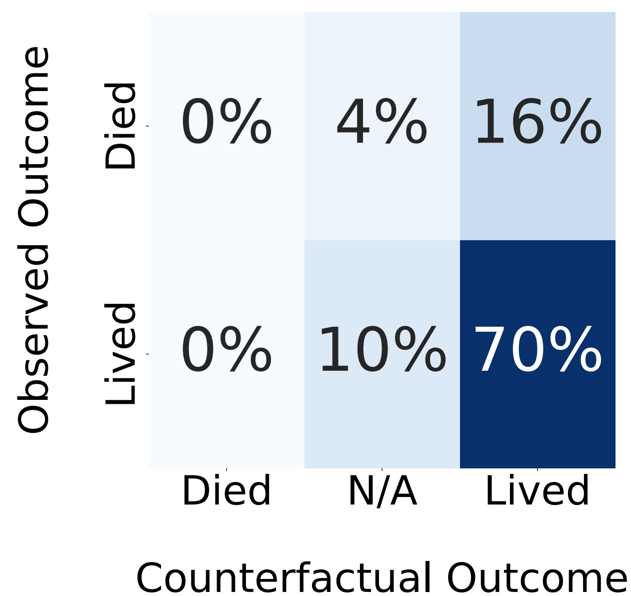
[Balke & Pearl, 1994]

# Using counterfactuals to "sanity check"

## Approach

1. **Decomposition of average reward** over real episodes, to identify interesting cases

## Example

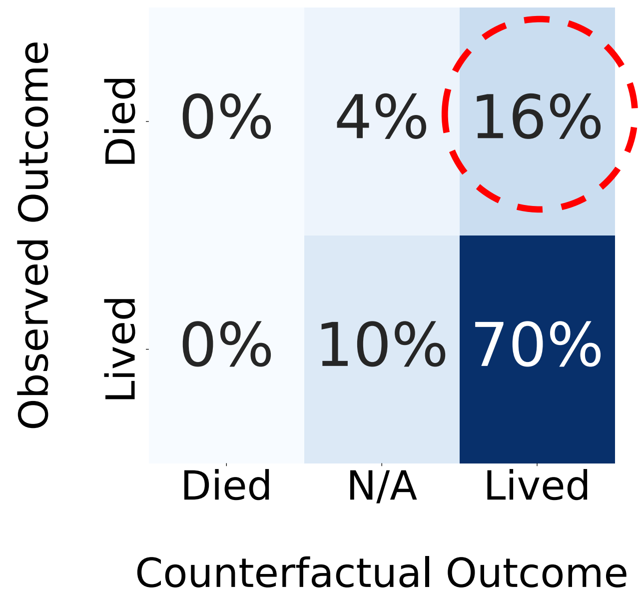


# Using counterfactuals to “sanity check”

## Approach

1. **Decomposition of average reward** over real episodes, to identify interesting cases
2. **Examine counterfactual trajectories** under new policy
3. **Validate and/or criticize** conclusions, using full patient information (e.g., chart review)

## Example



# Simulating counterfactual trajectories

**What we need**

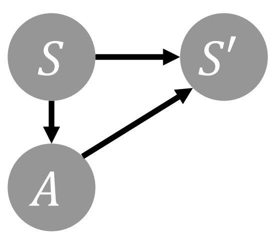
**1** Observed trajectories

**2** Policy to evaluate

$$\pi(A \mid S)$$

**3** Model of discrete dynamics, e.g., Markov Decision Process

$S$: Current State
$A$: Action
$S'$: Next State



# Simulating counterfactual trajectories

**What we need**

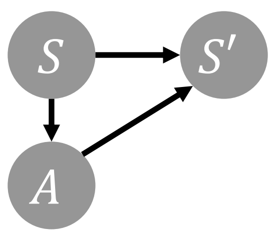
**1** Observed trajectories

**2** Policy to evaluate

$\pi(A \mid S)$

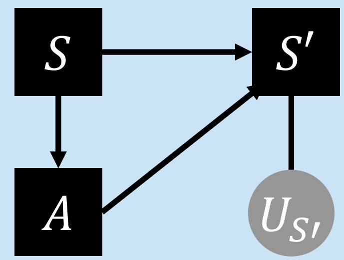
**3** Model of discrete dynamics, e.g., Markov Decision Process

$S$: Current State
$A$: Action
$S'$: Next State




**+**

**Structural Causal Model (SCM)**




$S' = f(S, A, U_{S'})$
$U_{S'} \sim P(U_{S'})$

# Simulating counterfactual trajectories

**What we need**

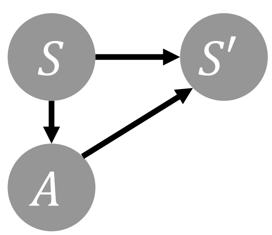
**1** Observed trajectories

**2** Policy to evaluate

$$\pi(A \mid S)$$

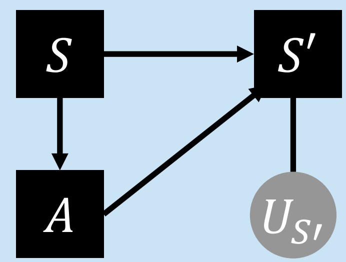
**3** Model of discrete dynamics, e.g., Markov Decision Process

$S$: Current State
$A$: Action
$S'$: Next State




**+**

**Structural Causal Model (SCM)**



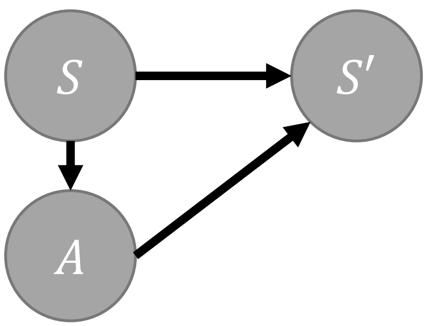

$$S' = f(S, A, U_{S'})$$
$$U_{S'} \sim P(U_{S'})$$

**Form of SCM is an *assumption***:
SCM is not identifiable from data!

# Structural Causal Models (SCMs)

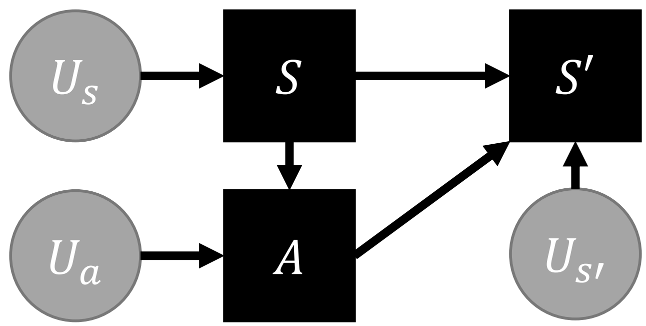
Use post-treatment information to reveal exogenous factors

**Causal Graph (Example)**




S, S’, A are R.V.s

**Structural Causal Model (SCM)**

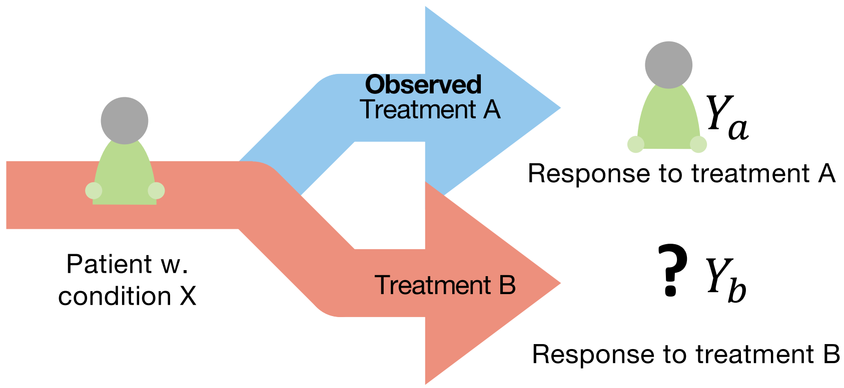



U’s are R.V.s / S, S’, A are functions

**Example:** $U_{s'} \sim Unif(0, 1)$,

$$S' = \begin{cases} 1, & U_{s'} \leq p \\ 0, & U_{s'} > p \end{cases} \quad \text{where } p \coloneqq \Pr[S' = 1 \mid S, A\,]$$

# Counterfactuals with SCMs

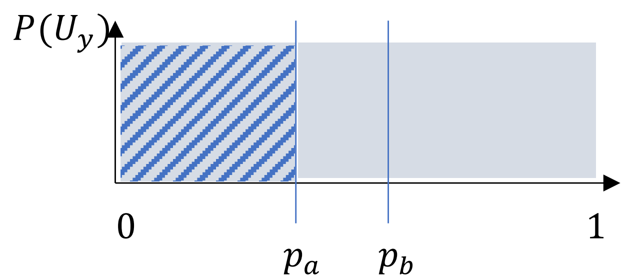



**Structural Causal Model**

$$U_y \sim Unif(0,1),$$

$$Y_t = \begin{cases} 1, & U_y \le p_t \\ 0, & U_y > p_t \end{cases} \quad \text{where } p_t := \Pr[Y = 1 \mid T = t, X]$$

1. **Infer** the posterior of $U_y$ given $X, Y_a = 1$
2. **Intervene** to set $T = b$
3. **Predict** counterfactual outcome $Y_b$

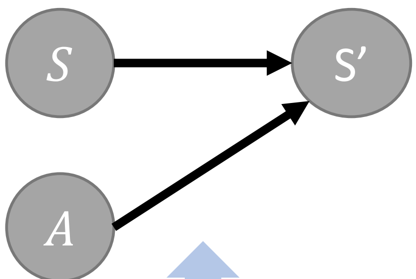



$$P\big(U_y \le p_b \,\big|\, U_y \le p_a\big) = 1$$
$$\to Y_b = 1$$

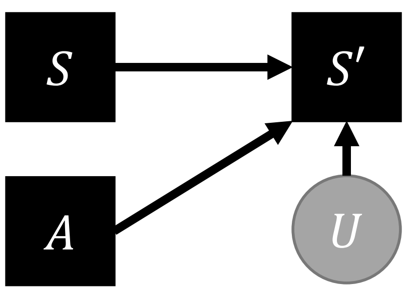
This SCM has the **monotonicity** *(Pearl 2000)* property, which implies that if $p_b \ge p_a$, then $Y_a = 1 \to Y_b = 1$

# SCMs for Markov Decision Processes

**Causal Graph (one step)**




**Structural Causal Model**




**Choosing a structural mechanism**

What is an appropriate SCM for categorical transitions?

$$p_{i|s,a} \coloneqq P(S' = i \mid S = s, A = a)$$

**Criteria 1:** Want to choose $f_s(S_t, A_t, U)$ and $\mathrm{P}(U)$ such that:

$$E_u[\, f_s(S_t = s, A_t = a, U) = i] = p_{i|s,a}$$

**Criteria 2:** Given unidentifiability of counterfactuals, want to make a "reasonable assumption" analogous to monotonicity (Pearl, 2000)

# Counterfactual Stability & Gumbel-Max SCM

## Counterfactual Stability

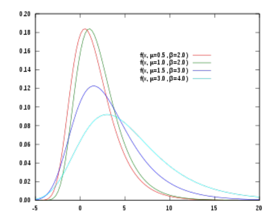
New *counterfactual stability* condition:

*If we observe* $S' = i$ *under* $A = a$*, then under counterfactual* $A = \tilde{a}$,

$$\frac{p_j}{p_i} > \frac{\widetilde{p_j}}{\widetilde{p_i}} \Rightarrow S' \neq j.$$

## Gumbel-Max SCM

Use the *Gumbel-Max trick* to sample from a categorical distribution with $k$ categories:



$$g_j \sim Gumbel$$
$$S' = argmax_j \, \{ \log P(S' = j \mid S, A) + g_j \}$$

**Theorem 1 (Oberst, Sontag 2019)**: Counterfactual stability implies monotonicity (Pearl, 2000) when $k = 2$

**Theorem 2 (Oberst, Sontag 2019)**: The Gumbel-Max SCM satisfies the Counterfactual Stability condition

# Summary

- Causal inference is a special case of off-policy reinforcement learning
- As a result, off-policy reinforcement learning is subject to the same assumptions:
  - Overlap
  - No unobserved confounding
- We suggested one approach of using **introspection** to help detect errors
- Much more work needed to get safe & robust algorithms