Decision-Making under Non-Stationarity: Concepts, Formulations, Algorithms, and Open Challenges

Ayan Mukhopadhyay
Assistant Professor of Computer Science
College of William & Mary

Nathaniel Keplinger
Ph.D. Student
Vanderbilt University

WILLIAM & MARY
Robust and Adaptive Intelligence Laboratory

VANDERBILT UNIVERSITY
scopelab.ai

Roadmap

- Why CPS researchers should care about decision-making under uncertainty?
- The big picture: MDPs in changing environments, and a fragmented literature.
- Stochastic control and Markov decision processes: the formal mathematical construct for decision-making.
- A unified problem formulation: the Parameterized Non-Stationary MDP.
- Deep dive: non-stationarity, formal definitions, and themes.
- How to tackle non-stationarity: the central role of collecting evidence about change.
- Hands-on session on using NS-Gym

VANDERBILT UNIVERSITY
scopelab.ai

Why does Uncertainty matter for CPS?

- A CPS is a tight loop: sense → decide → actuate → observe consequence → incorporate feedback .
- Almost all the components are affected by uncertainty.

Image Source: NLC

Image Source: SFCTA

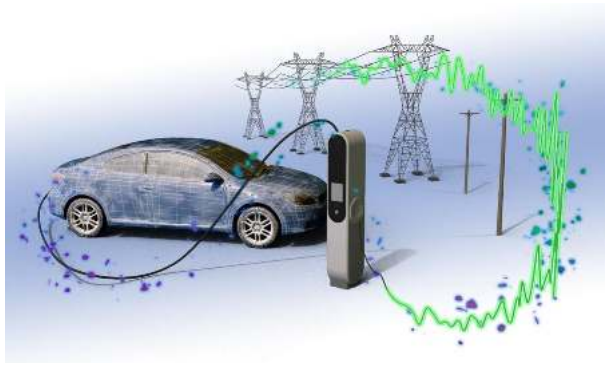


Image Source: ORNL

VANDERBILT UNIVERSITY
scopelab.ai

Where does Uncertainty arise?

- There are three main sources, all of them present in essentially every deployed CPS:
  - Sensing uncertainty. Noisy, biased, intermittent measurements.
  - Model uncertainty. Our analytical or learned model of the world is wrong in ways we cannot fully characterize.
  - Environmental uncertainty. The world itself is stochastic, e.g., weather, demand, human behavior, or potential adversaries.

Why do we need a principled framework?

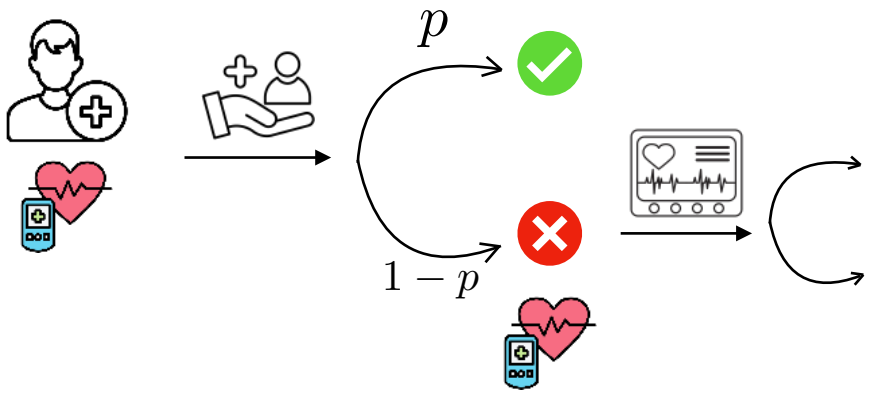
Ad-hoc heuristics work until they don't. We need a language that lets us reason about uncertainty, quantify robustness, and certify behavior.

VANDERBILT UNIVERSITY
scopelab.ai

From Uncertainty to Stochastic Control

Many real-world problems involve making sequential decisions over time under uncertainty

- Autonomous driving (Kiran et al. 2021)
- Medical diagnosis and treatment (Yu et al. 2021)
- Emergency response (Mukhopadhyay et al. 2022)
- Vehicle Routing (Li, Yan, and Wu 2021)
- Financial portfolio optimization (Pendharkar and Cusatis 2018)



VANDERBILT UNIVERSITY
scopelab.ai

Who makes the decision? Defining Agents

An *agent* in an entity capable of computation that acts or makes decisions based on observations from the environment.

The agent can be a human, an algorithm, or a combination of both.

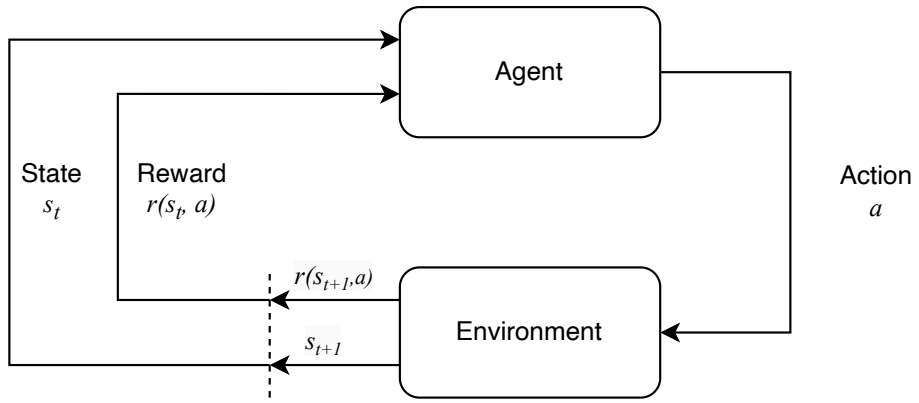
Modeling the Decision-Making Agent

Markov Decision Process: A Primer

A general way to model a sequential decision making problem under uncertainty is a **Markov Decision process (MDP).**

An **MDP** consists of the following components:

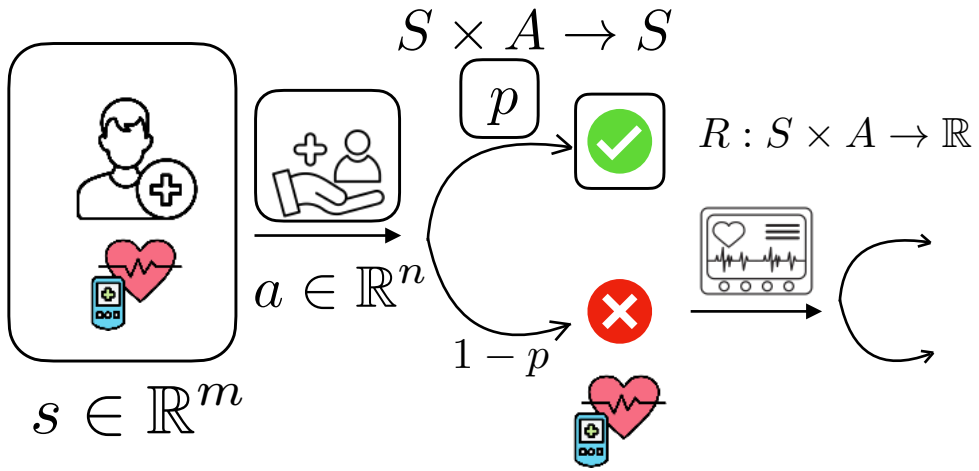
- **State ($\mathcal{S}$)**: This element summarizes our current "position" in the world or the current conditions of the system that we are interested in.
- **Actions ($\mathcal{A}$)**: The set of choices that the agent has access to. Note that all actions might not be available in all states.
- **Transition**: A probability distribution over the future state, given the current state and action, i.e., $P(s' \mid s, a)$.
- **Rewards**: A reward function, $R(s, a, s')$, that quantifies the instantaneous reward the agent gets by being in state s, taking action a and transitioning to state s'.



VANDERBILT UNIVERSITY
scopelab.ai

Markov Decision Process: A Primer

A general way to model a sequential decision making problem under uncertainty is a **Markov Decision process (MDP).**



An **MDP** consists of the following components:

- **State ($\mathcal{S}$)**: This element summarizes our current "position" in the world or the current conditions of the system that we are interested in.
- **Actions ($\mathcal{A}$)**: The set of choices that the agent has access to. Note that all actions might not be available in all states.
- **Transition**: A probability distribution over the future state, given the current state and action, i.e., $P(s' \mid s, a)$.
- **Rewards**: A reward function, $R(s, a, s')$, that quantifies the instantaneous reward the agent gets by being in state s, taking action a and transitioning to state s'.

VANDERBILT UNIVERSITY
scopelab.ai

Solving an MDP

- A **policy** $\pi : \mathcal{S} \to \mathcal{A}$ is a prescriptive solution for the agent.
- The **value** of a policy from state s is

$$V^{\pi}(s) \;=\; \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \, R(s_t, a_t, s_{t+1}) \,\middle|\, s_0 = s,\; a_t \sim \pi\right].$$

- The **action-value** (Q-function) is

$$Q^{\pi}(s, a) \;=\; \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \, R(s_t, a_t, s_{t+1}) \,\middle|\, s_0 = s,\; a_0 = a,\; a_{t\geq 1} \sim \pi\right].$$

- An optimal policy π^* maximizes the expected discounted set of rewards.
- A remarkable property of using **discounted rewards** with infinite horizon is that π^* is independent of the starting state; so we can simply refer to the optimal policy as π^*.

VANDERBILT UNIVERSITY
scopelab.ai

The Bottleneck: Computing An Optimal Policy

- Let us start by writing the "utility" of being in a state:

$$U(s) = \max_{a \in A_s} \sum_{s'} P(s' \mid s, a)[r(s, a, s') + \gamma U(s')]$$

- There are n equations and n unknowns—the utilities of a state.
- We can simply solve these simultaneous equations to find the utilities.
- What is the issue?
- There is the pesky max operator.

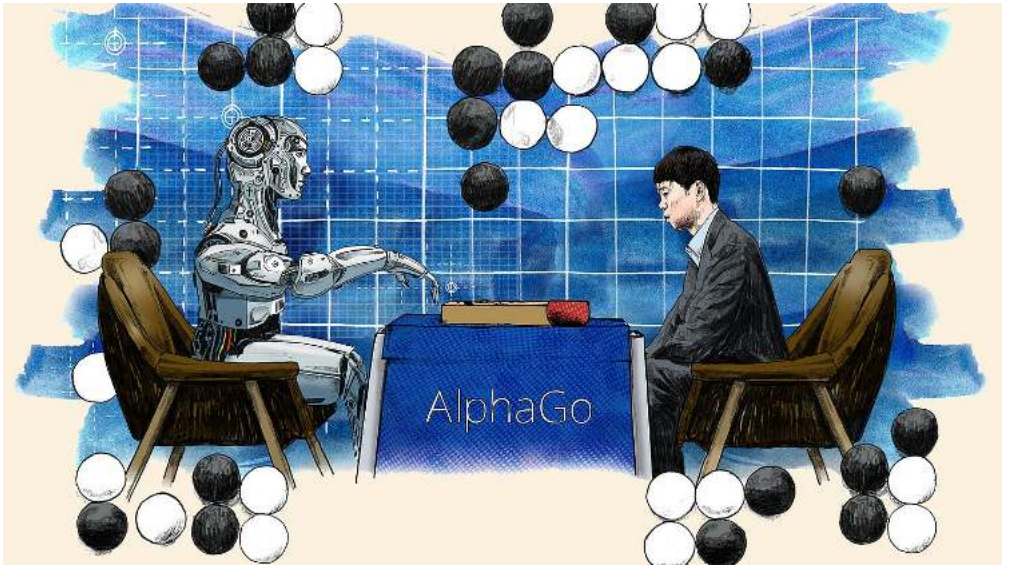
Enter Richard Bellman

- Let us try an iterative approach.
  - Let's initialize the utilities with arbitrary values.
  - Given the arbitrary utility values, we can compute the right hand side of the equations.
  - This gives us new values for the utilities.
  - Can this possibly be good?
  - It is, in fact, great!!!
  - Simply doing this ensures convergence to the optimal utilities.

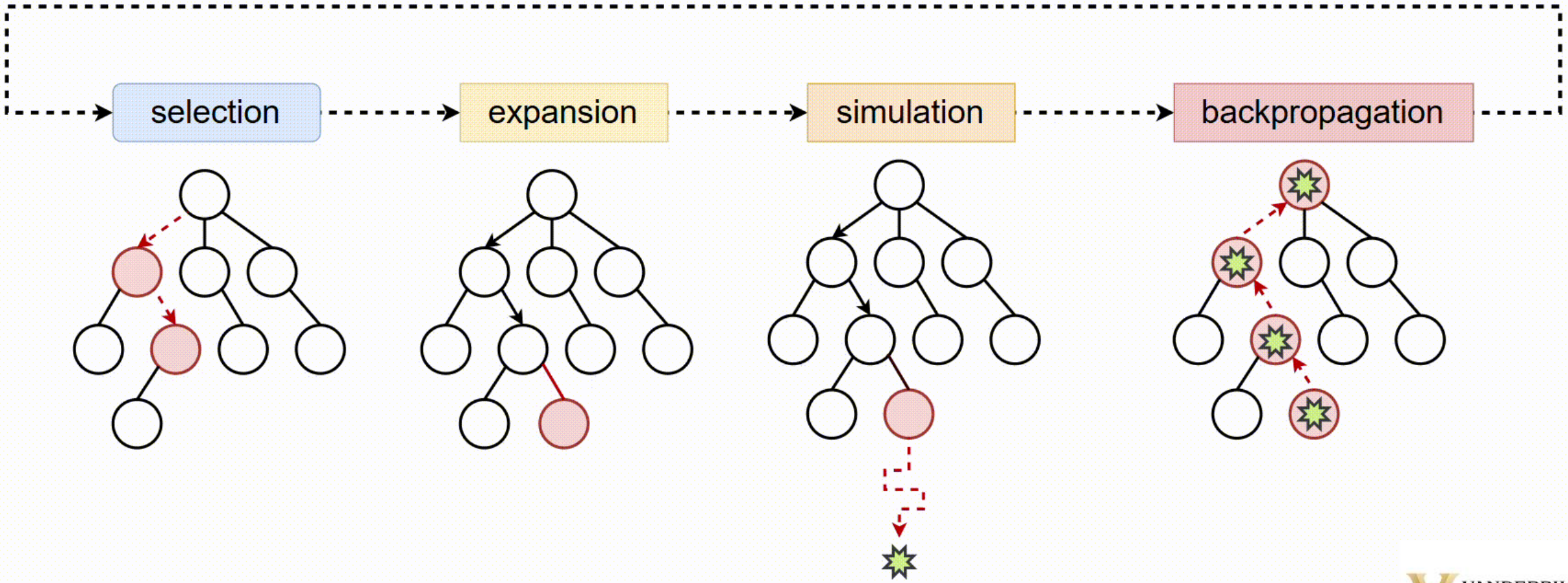
VANDERBILT UNIVERSITY
scopelab.ai

Can We Claim Victory? (1)

- It turns out that we cannot use dynamic programming for every problem, since even though it can be completed in polynomial time, its time complexity rises quadratically in the number of states.
- Instead of finding a general (and preferably optimal) mapping from states to actions, we can instead compute the optimal way to act in the current state.
- This argument is the basis of online planning, e.g., approaches like Monte Carlo Tree Search. It formed the backbone of the famous AlphaGo framework from DeepMind.



Monte Carlo Tree Search



MDPs in Changing Environments

Let us Consider an Example

- A Running Example: A Robotic Arm. Imagine a robotic arm trained to sort objects on a conveyor belt.
  - State $\mathcal{S}$: joint angles, gripper status, position of the next object.
  - Actions $\mathcal{A}$: motor torques and gripper open/close.
  - Transition P: rigid-body dynamics of the arm and the conveyor.
  - Reward R: +1 for each object placed in the correct bin.

- At deployment, three things start happening on different timescales:
  - Joints wear out, a slow drift in P.
  - New product lines bring new object shapes and weights, the initial-state distribution shifts.
  - The operator reassigns the task: sort by fragility, not size, an abrupt change in R.

- The MDP we wrote down at training time no longer describes the world the arm operates in. We will keep returning to this arm.

VANDERBILT UNIVERSITY
scopelab.ai

A Fragmented Literature

Sequential decision making in changing environments has been studied for decades, under many different names, by many different sub-communities:

- **Non-stationary MDPs:** the world drifts with time and the agent must adapt at decision time.
- **Contextual MDPs (C-MDPs):** a latent or observed context defines the dynamics.
- **Hidden-Parameter MDPs (HiP-MDPs):** the context is latent and must be inferred.
- **Meta-reinforcement learning:** learn to adapt across a distribution of MDPs.
- **Robust MDPs:** optimize for the worst case over a set of possible MDPs.
- **Continual / lifelong RL:** a stream of tasks, without forgetting.

Different Lenses for Related Problems

The Arm Through the Non-Stationary MDP Lens

Core intuition: The transition function P_t shifts at (potentially) every step, and the agent must adapt (this is the focus of our tutorial).

- Consider gradual torque degradation or a sudden loss of one robot in a multi-robot environment.
- The previously optimal policy becomes progressively suboptimal.
- Formalized as $\{P_t, R_t\}, t \geq 0$, i.e., varying with time.
- The change must first be detected and then the agent adapt.

The Arm Through the Contextual / HiP-MDP Lens

Core intuition: The world is not really changing, but revealing different *forms* of a fixed underlying structure.

- Consider each batch of objects has a different weight distribution, and the arm's wear level is set at beginning of a task
- Each task is internally stationary; only the parameter varies across tasks.

VANDERBILT UNIVERSITY
scopelab.ai

Different Lenses for Closely Related Problems

The Arm Through the Meta-RL Lens.

Core Intuition: *Learn* to adapt.

- If the arm has trained on many wear levels and object distributions, leave behind a learning algorithm that adjusts after a few interactions.
- Blind spot: the test task must be drawn from the training distribution

The Arm Through the Robust MDP Lens.

Core Intuition: Don't track the change, instead, be safe against all of them.

- Optimize $\max_{\pi} \min_{P \in U} V_P^{\pi}$ over an uncertainty set U.
- Blind spot: conservatism. The minimax can sacrifice average performance to hedge against unlikely worst cases.

The Arm Through the Continual / lifelong RL Lens.

Core Intuition: Sequential tasks without forgetting.

- The arm should retain motor skills when reassigned from size-sorting to fragility-sorting.
- Blind spot: does not focus on data collection and immediate adaptation.

VANDERBILT UNIVERSITY
scopelab.ai

An Unifying Formalism

- We argue that the formalisms in Part 3 are not fundamentally distinct. They are special cases of a single parent model, distinguished along two orthogonal axes:
  - **The dynamics of the context.**
    - How does the latent parameter governing the environment evolve?
    - Fixed? IID across episodes?
    - Slowly drifting? Adversarially chosen?
  - **The observability of the context.**
    - What does the agent know about the current parameter?
    - Observed at episode start?
    - Fully hidden?
    - Partially revealed through the trajectory?
- Recognizing this structure lets us import tools and results across sub-communities.

VANDERBILT UNIVERSITY
scopelab.ai

P-NS-MDP

- The Parameterized Non-Stationary MDP (PNSMDP)
- **Definition** A PNS-MDP is a tuple $\mathscr{M} = \left(\mathcal{S}, \mathcal{A}, \mathscr{C}, P, R, q, \mu_0^s, \mu_0^c, \right)$, where
  - $\mathscr{C}$ is a context space,
  - $P : \mathcal{S} \times \mathcal{A} \times \mathscr{C} \to \Delta(\mathcal{S})$ is a context-conditioned transition function.
  - $R : \mathcal{S} \times \mathcal{A} \times \mathscr{C} \to \mathbb{R}$ is a context-conditioned reward function.
  - $q : \mathscr{C} \to \Delta(\mathscr{C})$ is a context dynamics kernel.
- **Generative process.** Draw $s_0 \sim \mu_0^s$ and $c_0 \sim \mu_0^c$. Then
- The stochastic process evolves as: $s_{t+1} \sim P(\cdot \mid s_t, a_t, c_t), \quad c_{t+1} \sim q(\cdot \mid c_t)$.
- The kernel q is the central object that distinguishes the sub-fields.
- Different structural assumptions on $q$give qualitatively different problems.

VANDERBILT UNIVERSITY
scopelab.ai

Everything Fits into this Meta-Lens

Formalism	Context dynamics q	c_t observed?	Within-deployment change?	Agent's challenge
Standard MDP	$\mathcal{C} = \{c_0\}$, trivial	Irrelevant	No	Find an optimal policy
Contextual MDP	i.i.d. across deployments	Yes	No	Generalize across contexts
HiP-MDP	i.i.d. across deployments	No (latent)	No	Infer c from trajectory
Meta-RL	i.i.d. across deployments	No (latent)	No	Learn to adapt quickly
Non-stationary MDP	c_t undergoes bounded change	No	Yes	Track drift online
Robust MDP†	Adversarial $c \in \mathcal{U}$	No	Yes	Worst-case performance

Five seemingly distinct formalisms, all recovered by varying just two things: the structure of q and whether c_t is observed.

VANDERBILT UNIVERSITY
scopelab.ai

Non-Stationary MDPs

Diving Deep into Non-Stationarity

- We now have a unified picture of MDPs in changing environments.
- Each cell of the table is its own research area, with its own algorithms and guarantees.
- For the rest of this tutorial, we zoom in on the NS-MDP cell: the change is unexpected and the agent must adapt online.
- Why this cell? Because it is the cell most CPS deployments actually live in — and the cell where the open problems are most acute.

VANDERBILT UNIVERSITY
scopelab.ai

How Brittle is AI-driven CPS?

Unfortunately, the real world is not stationary.

What is a stationary stochastic process?

- A stochastic process whose unconditional joint probability distribution does not change when shifted in time.
- Formally, let $\{X_t\}$ be a stochastic process and let $F_X(x_{t_1+\tau}, \ldots, x_{t_n+\tau})$ represent the cumulative distribution function of the marginal distribution (i.e., with no reference to any particular starting value) of $\{X_t\}$ at times $t_1 + \tau, \ldots, t_n + \tau$.
- Then, $\{X_t\}$ is said to be "strictly stationary", if

$F_X(x_{t_1+\tau}, \ldots, x_{t_n+\tau}) = F_X(x_{t_1}, \ldots, x_{t_n})$ for all $\tau, t_1, \ldots, t_n \in \mathbb{R}$ and for all $n \in \mathbb{N}_{>0}$

VANDERBILT UNIVERSITY
scopelab.ai

A Simple Discrete State Discrete Action MDP

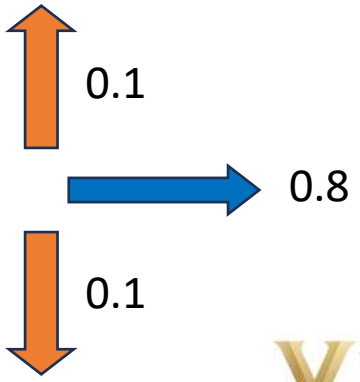
Let us look at a simple example. Consider the Frozen Lake environment from OpenAI Gym.

State: The location of the agent, holes, and goal.

Actions: Go up, right, left, or down.

Transitions: The agent can transition to a new cell corresponding to an action with some chance of transitioning to an adjacent cell instead.

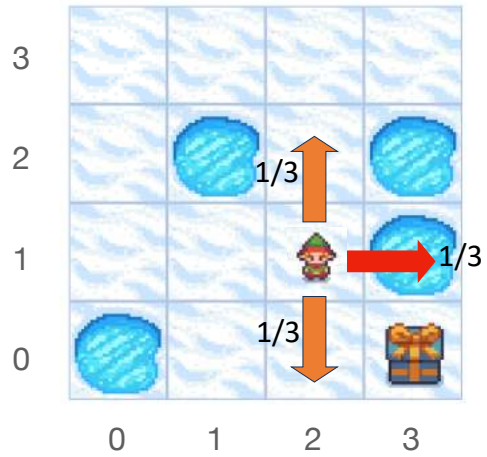
Rewards: There is a positive reward if we reach the goal.



VANDERBILT UNIVERSITY
scopelab.ai

How can we Quantify Non-Stationarity?

- Consider the transition probability function $P_t(s' \mid s, a)$, where the subscript t denotes the time step under consideration. Now, consider that the environment undergoes *some* change between time steps 0 and t.



$$\forall s, a : \sum_{s' \in S} |P_t(s' \mid a, s) - P_0(s' \mid a, s)| \leq \eta$$

$Q_t^{\pi}(s, a)$ The expected discounted sum of all future rewards when starting from state s, taking action a, and following policy π thereafter.

Theorem 1: If $\forall s, a : \sum_{s' \in S} \left| P_t(s' \mid a, s) - P_0(s' \mid a, s) \right| \leq \eta$, and $\forall s, a : |r(s, a)| \leq R$, and the discount factor $\gamma < 1$, then

$$|Q_0^{\pi_0^*}(s, a) - Q_t^{\pi_t^*}(s, a)| \leq \frac{\gamma \cdot \eta \cdot R}{(1-\gamma)^2}; \forall s, a$$

ϵ

Expected Long-Term Rewards by taking action a and following the optimal policy at time 0

Expected Long-Term Rewards by taking action a and following the optimal policy at time t

VANDERBILT UNIVERSITY
scopelab.ai

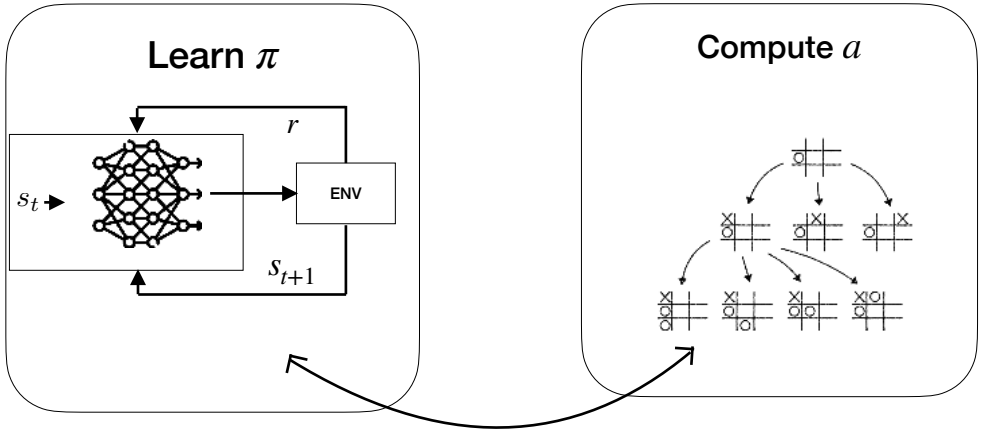
How Can we Make Decisions?


Ava Pettet


Yunuo Zhang

- Our goal is to find optimal actions at execution time t under the following conditions:
  - We are given an optimal action-value function $Q_0^{\pi_0^*}(s,a)$ for time step 0 but not at execution time t.
  - The environmental change is bounded by η.
  - For now, we assume that the agent "knows" the environmental model at time t.
  - There is a limited computational budget at execution time that prevents re-learning an optimal policy.

Policy-Augmented Monte Carlo Tree Search (PA-MCTS)



Key Insight: If the world has not changed much, should we throw away the knowledge about how to act?

$$\text{argmax}_{a \in \mathcal{A}_s} \alpha Q_0^{\pi_0^*}(s,a) + (1-\alpha)\overline{G}_t(s,a)$$

Existing knowledge, optimal in the original environment

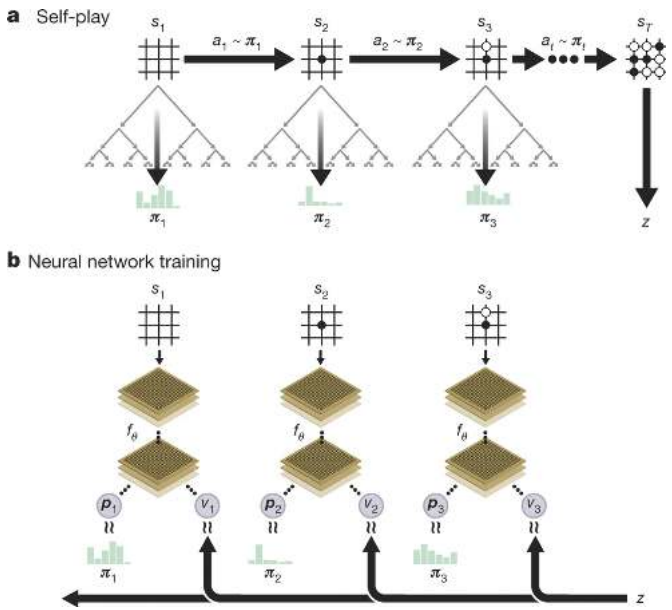
Should the agent trust the old knowledge or the new computation?

Online computation in the new environment

VANDERBILT UNIVERSITY

scopelab.ai

Déjà vu



- Key Differences:
  - Combining an existing policy and online search “outside the tree” stabilizes search under non-stationarity.
  - This stabilization also allows significantly faster convergence than AlphaZero.
  - We show experimentally that this combination comprehensively outperforms AlphaZero in multiple domains.
  - Unlike AlphaZero, we present strong theoretical guarantees!!

Image Source: Silver et al., Mastering the game of Go without human knowledge. *Nature* (2017)

VANDERBILT UNIVERSITY
scopelab.ai

How well does this work?

DDQN

Current State: [0, 0]
Next State: [0, 0]
Decision: down
Decision Q Value Estimate: 0.0

Success Rate: 0.12 ± 0.01

AlphaZero

Current State: [0, 0]
Next State: [0, 0]
Decision: down
Decision Q Value Estimate: 0.0

Success Rate: 0.114 ± 0.011

MCTS

Current State: [0, 0]
Next State: [0, 0]
Decision: down
Decision Q Value Estimate: 0.0

Success Rate: 0.866 ± 0.01

Policy-Augmented Search

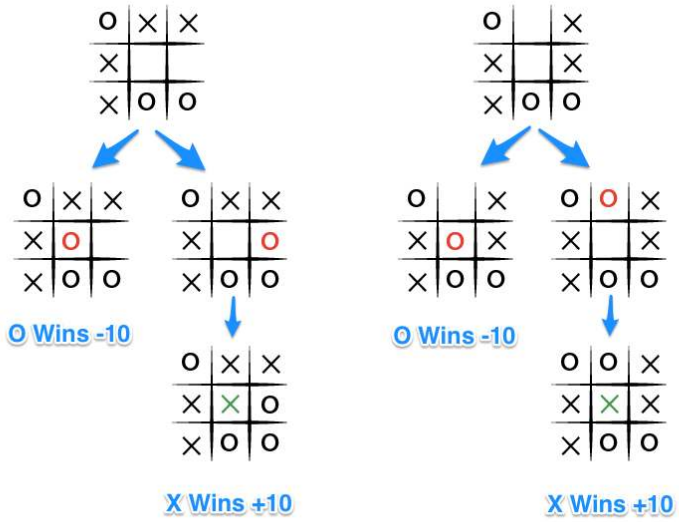
Current State: [0, 0]
Next State: [0, 0]
Decision: down
Decision Q Value Estimate: 0.0

Success Rate: 0.936 ± 0.009

VANDERBILT UNIVERSITY
scopelab.ai

Can we claim victory? (2)

- We assumed that the agent "knows" the updated environment.
- The idea of a *snapshot model* is widely accepted in the community, e.g., Lecarpentier and Rachelson (2019) proposed a fully online approach to tackle non-stationarity based on a snapshot model.
- However, can data collection and decision-making be decoupled?



We challenged these assumptions

- We argue that updated **transition dynamics are not known** to the agent.
- We argue that the **lack of knowledge about non-stationarity is not monolithic**.
- As the agent interacts with the environment, there are regions of the state space that it learns more about than other regions.

The worst case evolution of the environment

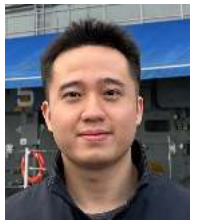
$$Q^*(s,a) := \min_{\hat{p}} \mathbb{E}_{s' \sim \hat{p}} \left[R(s,a) + \gamma V^*_{t+1}(s') \right]$$

How can the agent respond?

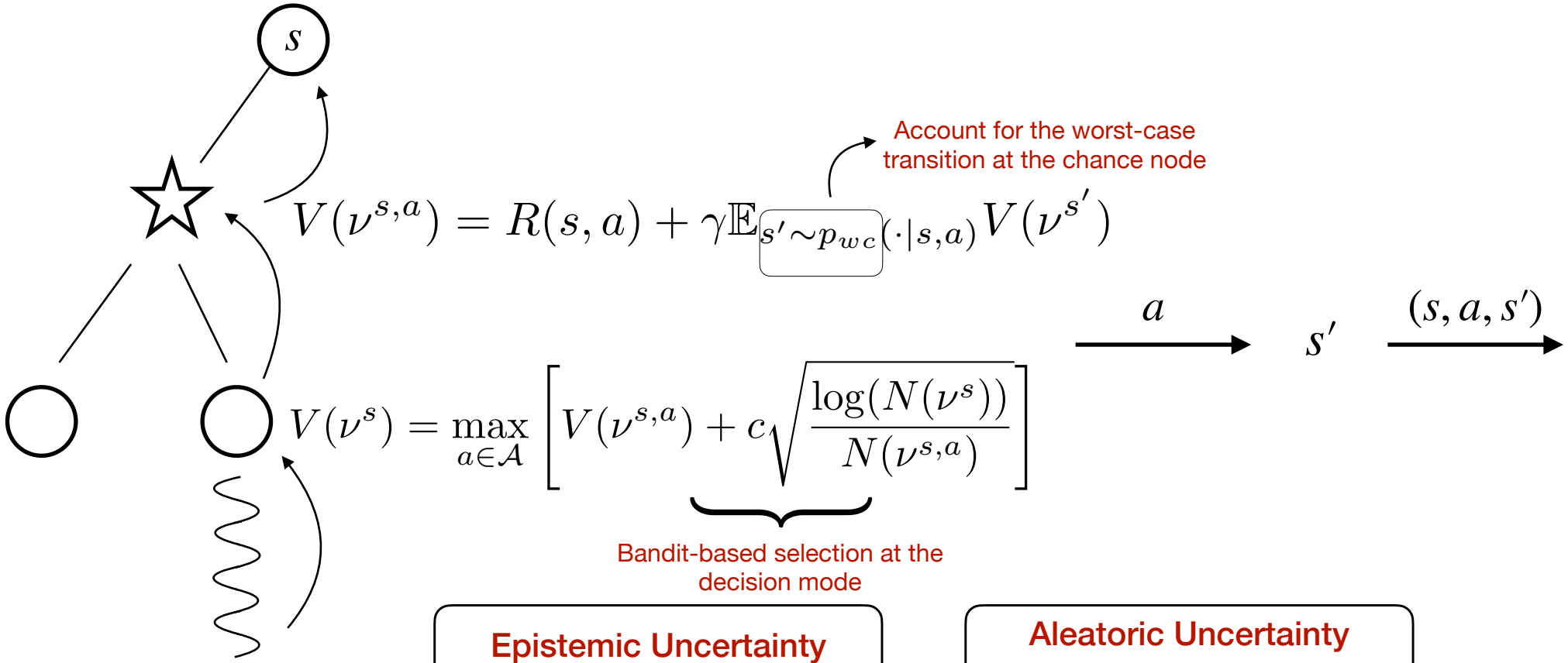
$$V^*(s) := \max_{a \in \mathcal{A}} Q^*(s,a)$$

VANDERBILT UNIVERSITY
scopelab.ai

Fully Adaptive Decision Making (Ada-MCTS)



Baiting Luo



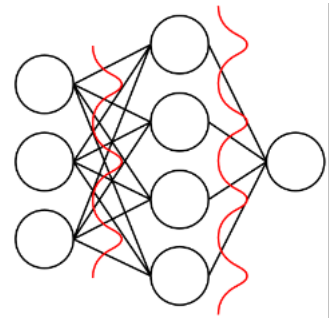
$$V(\nu^{s,a}) = R(s,a) + \gamma \mathbb{E}_{s' \sim p_{wc}(\cdot|s,a)} V(\nu^{s'})$$

Account for the worst-case transition at the chance node

$$V(\nu^{s}) = \max_{a \in \mathcal{A}} \left[V(\nu^{s,a}) + c\sqrt{\frac{\log(N(\nu^{s}))}{N(\nu^{s,a})}} \right]$$

Bandit-based selection at the decision mode

a → s' → (s, a, s')



Epistemic Uncertainty

gauge the model's familiarity with the state-action space

Aleatoric Uncertainty

assess the randomness off the environment

VANDERBILT UNIVERSITY
scopelab.ai

How well does this work?

Problem Environment

Changing Environment Settings
(Higher means more deterministic transitions)

Monte Carlo Tree Search

Risk Averse Tree Search

Environment	Setting	Approaches that know ground truth transition		Approaches that do not know ground truth transition			
		MCTS-P_k	RATS-P_k	MCTS-$\hat{P}_{k-1}$	RATS-P_{k-1}	RATS-$\hat{P}_{k-1}$	ADA-MCTS
Cliff Walking	0.4	0.629 ± 0.11	0.650 ± 0.04	-0.518 ± 0.17	0.079 ± 0.17	0.630 ± 0.04	**0.778 ± 0.02**
	0.5	0.332 ± 0.18	0.605 ± 0.03	-0.304 ± 0.20	0.384 ± 0.15	0.664 ± 0.04	**0.815 ± 0.02**
	0.6	0.342 ± 0.18	0.670 ± 0.04	-0.292 ± 0.20	0.253 ± 0.15	0.607 ± 0.04	**0.830 ± 0.01**
	0.8	0.464 ± 0.18	0.647 ± 0.08	0.279 ± 0.20	0.625 ± 0.08	0.680 ± 0.04	**0.871 ± 0.01**
	0.9	0.564 ± 0.17	0.614 ± 0.07	0.561 ± 0.17	0.758 ± 0.02	0.622 ± 0.04	**0.883 ± 0.01**
	1.0	0.951 ± 0.00	0.750 ± 0.04	**0.947 ± 0.00**	0.000 ± 0.00	0.000 ± 0.00	0.694 ± 0.00
NS Bridge	0.4	-0.697 ± 0.02	-0.660 ± 0.03	-0.707 ± 0.03	-0.684 ± 0.03	-0.684 ± 0.03	**-0.642 ± 0.03**
	0.5	-0.436 ± 0.13	-0.624 ± 0.03	-0.499 ± 0.12	-0.590 ± 0.05	-0.590 ± 0.05	**-0.499 ± 0.08**
	0.6	-0.458 ± 0.13	-0.563 ± 0.04	**-0.384 ± 0.14**	-0.512 ± 0.05	-0.512 ± 0.05	-0.429 ± 0.09
	0.8	0.030 ± 0.16	-0.256 ± 0.05	-0.082 ± 0.16	-0.206 ± 0.06	-0.206 ± 0.06	**-0.075 ± 0.10**
	0.9	0.245 ± 0.15	-0.071 ± 0.04	**0.135 ± 0.15**	-0.044 ± 0.03	-0.044 ± 0.03	0.109 ± 0.08
	1.0	0.729 ± 0.00	0.000 ± 0.00	**0.729 ± 0.00**	0.000 ± 0.00	0.000 ± 0.00	0.183 ± 0.02
Frozen Lake	0.4	-0.882 ± 0.10	0.401 ± 0.16	-0.892 ± 0.10	0.264 ± 0.19	0.225 ± 0.18	**0.426 ± 0.12**
	0.5	-0.695 ± 0.16	0.422 ± 0.16	-0.695 ± 0.16	0.239 ± 0.18	0.336 ± 0.17	**0.446 ± 0.13**
	0.6	-0.497 ± 0.19	0.370 ± 0.15	-0.302 ± 0.21	**0.528 ± 0.14**	0.500 ± 0.13	0.474 ± 0.07
	0.8	0.491 ± 0.19	0.485 ± 0.09	-0.003 ± 0.22	0.514 ± 0.06	0.416 ± 0.09	**0.516 ± 0.11**
	0.9	0.689 ± 0.16	0.146 ± 0.07	0.392 ± 0.20	0.169 ± 0.05	0.169 ± 0.05	**0.490 ± 0.12**
	1.0	0.988 ± 0.00	0.889 ± 0.02	**0.988 ± 0.00**	0.000 ± 0.00	0.000 ± 0.00	0.782 ± 0.02

Expected Discounted Returns across 1000 runs

In general, adaptive MCTS produces significantly higher utility

If the world is deterministic, being careful is not optimal

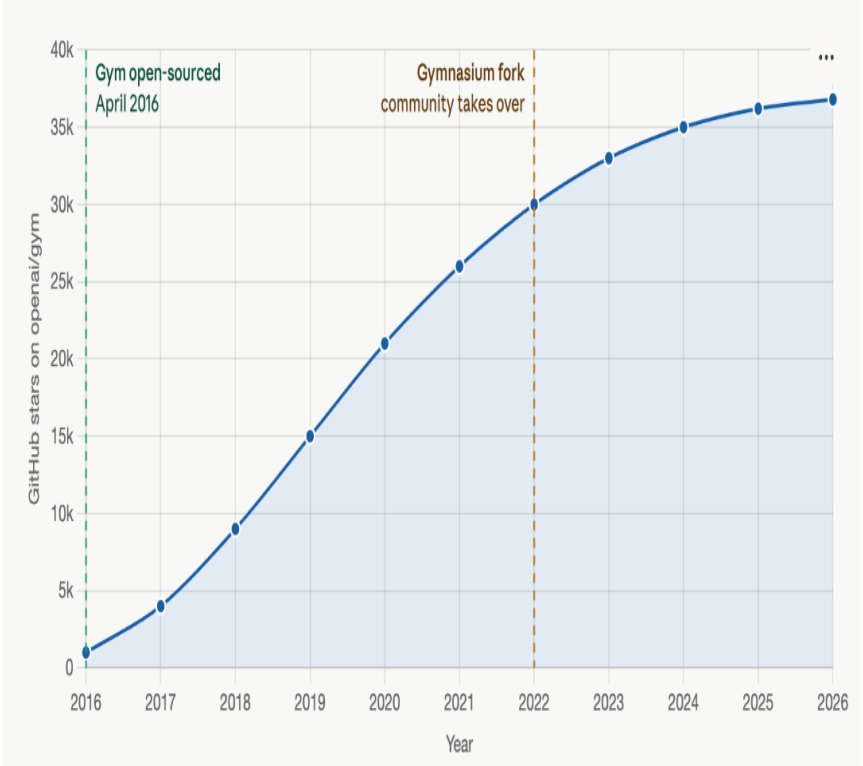
VANDERBILT UNIVERSITY
scopelab.ai

Enabling the Community to Work on Non-Stationary Stochastic Control

VANDERBILT UNIVERSITY
scopelab.ai

The History of Non-Stationarity in Decision Theory

- Non-stationarity has been well explored from the decision-making and control perspectives.
- Unfortunately, decision-theoretic methods have not adopted any unifying theme.
- Is a unifying *theme* critical?
  - Perhaps not, heterogeneous definitions are critical as application domains have diverse requirements.
  - However, a general mathematical model is important for propelling the field.
  - General mathematical models can also be used for develop standardized benchmarks.



VANDERBILT UNIVERSITY
scopelab.ai

The History of Non-Stationarity in Decision Theory

Early Conceptualization:

- Switching Environments (Ackerson & Fu, 1970):
  - Mean and covariance change over time.

Formalizing Non-Stationarity:

- Sojourn-Time-Dependent Markov Chains (Campo, Mookerjee, & Bar-Shalom, 1991):
  - Semi-Markovian processes with parameter changes after random intervals.

Recent Developments:

- Adapting to a Single Change:
  - Observed changes (Pettet and Zhang et al., 2024) or unobserved changes (Luo et al., 2024).
- Continuous Parameter Changes:
  - Modeled by Lecarpentier & Rachelson (2019).
- Forecast-Based Optimization (Chandak et al., 2020):
  - Maximize future policy performance instead of directly modeling non-stationarity.

VANDERBILT UNIVERSITY
scopelab.ai

Identifying the Key Questions

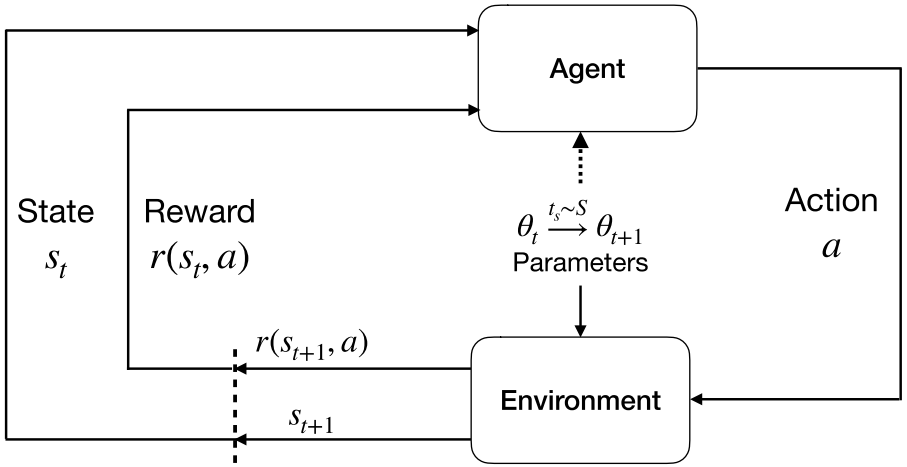
There are four questions of importance:

1. **What** changes?
2. **How** does it change?
3. Does the agent **detect** the change?
4. Does the agent **know** the change?

Putting Prior Work in Context

Model	Reference	Is the change notified?	Is the change known?	What changes?	Nature of the change	Is the change bounded?
Piecewise Stationary MAB	Garivier and Moulines (2011)	No	No	Reward	The reward distribution is fixed over certain time periods, and then changes at unknown time steps.	No
Non-stationary MAB	Besbes, Gur, and Zeevi (2014)	No	No	Reward	The reward can change at arbitrary time points.	Yes
Piecewise Stationary MDP	Auer, Jaksch, and Ortner (2008)	No	No	Transition, Reward	Bounded change analyzed as part of the UCRL2 algorithm	Yes
Non-Stationary MDP	Cheung, Simchi-Levi, and Zhu (2020)	N/A	No	Transition, Reward	The reward and transition can change at every time step	Yes
Non-Stationary MDP	Chandak et al. (2020b)	Yes	No	Transition, Reward	Transition and reward can change after each episode, but remain fixed within an episode	No
Non-Stationary MDP	Chandak et al. (2020a)	Yes	No	Transition, Reward	Transition and reward can change after each episode, but remain fixed within an episode	Yes
Non-Stationary MDP	Lecarpentier and Rachelson (2019)	Yes	Yes	Transition	The agent knows the current parameters, but not the future evolution.	Yes
Non-Stationary MDP	Pettet et al. (2024)	Yes	Yes	Transition	A single discrete change	Yes
Non-Stationary MDP	Luo et al. (2024)	Yes	No	Transition	A single discrete change	No
Non-Stationary Bandits with Periodic Variation	Chakraborty and Shettiwar (2024)	No	No	Reward	Periodic Variation	Yes

A General Interface and Toolkit for Non-Stationary MDPs



Standing on the shoulders of seminal work

- The Campo et al. paper from 1991 presents a very elegant conceptualization—the Markovian model for the endogenous and the exogenous uncertainties can (and should) be disentangled.
- We adopt their conceptualization.
- A “base MDP” can be used for the underlying control process.
- A semi-Markov process can be used for environmental parameters that change.

VANDERBILT UNIVERSITY
scopelab.ai

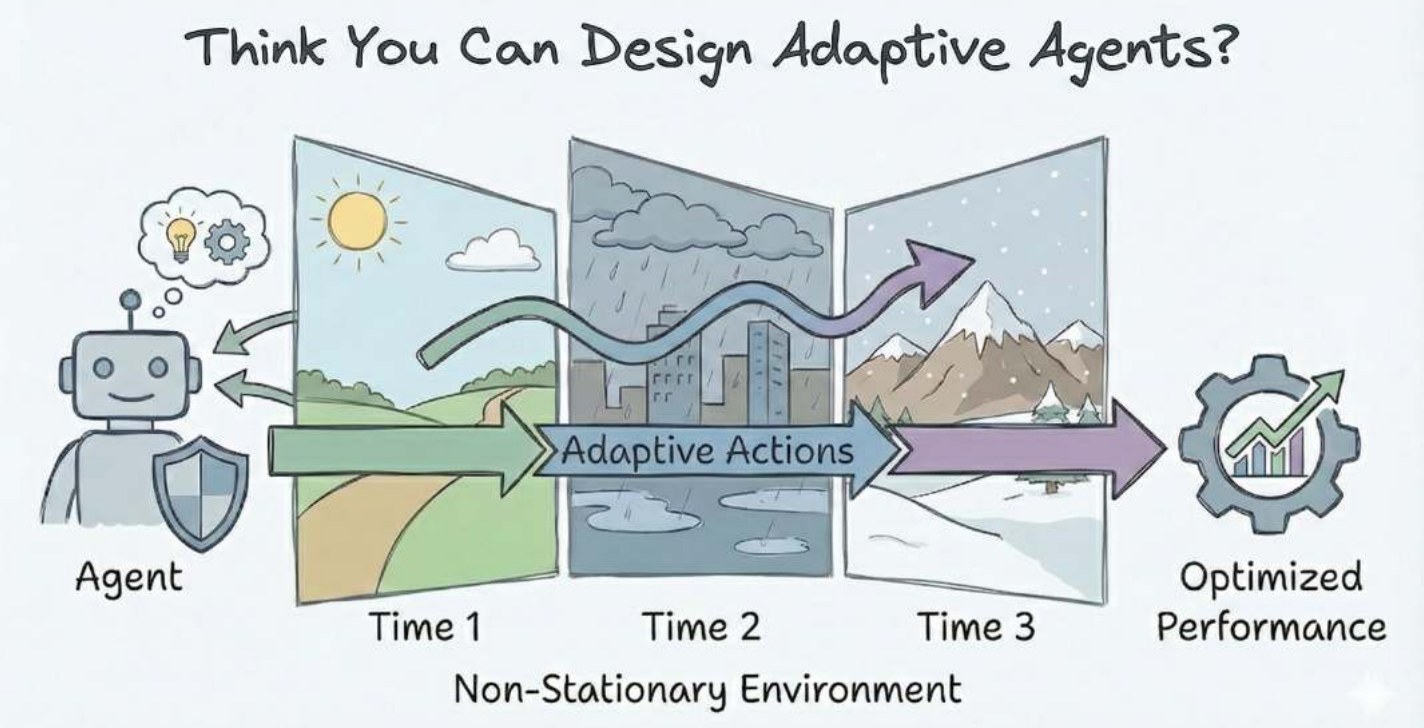
NS-Gym

NS-Gym is a simulation toolkit for NS-MDPs that provides a tailored, standardized, and principled set of interfaces for non-stationary environments.

1. To our knowledge, there are no standard problems or benchmarks for NS-MDPs.
2. A toolkit is needed that captures canonical problem instances of decision making in NS-MDPs.
3. We need a simulation framework that offers broad NS-MDP specifications while maintaining an easy-to-use, familiar interface for researchers.

VANDERBILT UNIVERSITY
scopelab.ai

AAMAS Competition on Non-Stationarity



Learn to design AI agents & compete at the premier CS conference on multi-agent systems.

- **Feb 5, 2026**: Baseline Competition Code and Tutorial Release
- **January - April 2026**: Submission period, Q&A forum, and Office Hours
- **May 10, 2026**: Final Submission Deadline
- **May 25-29, 2026**: AAMAS 2026 Conference (Paphos, Cyprus)

SCAN ME

2026 AAMAS PAPHOS • CYPRUS 25 YEARS

VANDERBILT UNIVERSITY
scopelab.ai

Collaborators




This work is based on NSF Award CNS-2238815, CNS-2531369 and the *Assured Neuro Symbolic Learning and Reasoning (ANSR)* project sponsored by the Defense Advanced Research Projects Agency (DARPA)