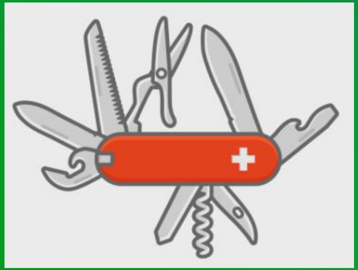
Is Reinforcement Learning all you need?

An algorithm cheat sheet for sequential decision making with applications to telecom

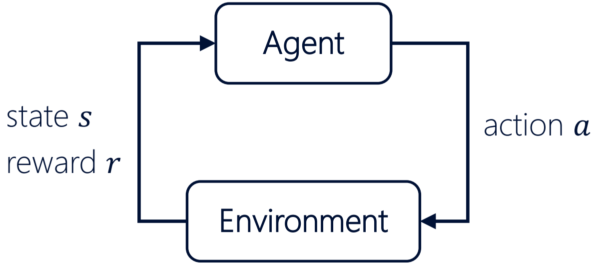
Lorenzo Maggi (Nokia Bell Labs)



Sequential planning

Markov Decision Process model

- An **agent** (in telecom: base station, SON server, UE)
  - ✓ observes the current "**state**" s_t of the environment
    - In telecom: channel conditions, UE buffer state, past UE throughput
  - ✓ takes an "**action**" a_t according to a probabilistic strategy π
    - In telecom: beam coefficients, which UE to schedule, which cell to switch off
  - ✓ receives a "**reward**" r_t
    - In telecom: UE throughput, energy savings



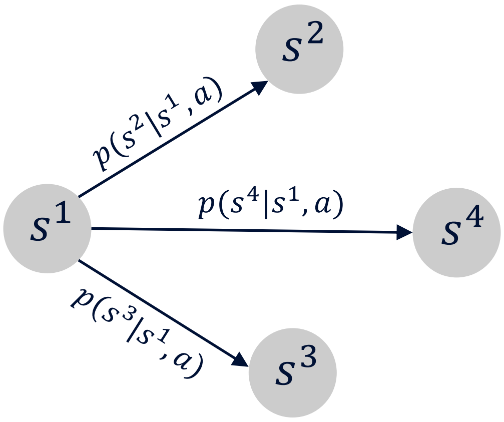
- The **environment** (in telecom: neighbor BTSs/UEs) reacts to the agent's action and
  - ✓ changes its "state" according to some **stochastic law** $p(s_{t+1}|s_t, a_t)$

- Agent's **goal**: maximize sum of rewards across time $\max_{\pi} \sum_t \beta^t r(s_t, a_t)\,, \beta \in [0,1)$
- Agent *may* be **oblivious** of such model and only observe new states/rewards

Sequential planning

Markov Decision Process model

- Take action a from distribution $\pi(s^1)$
- Receive reward $r(s^1, a)$



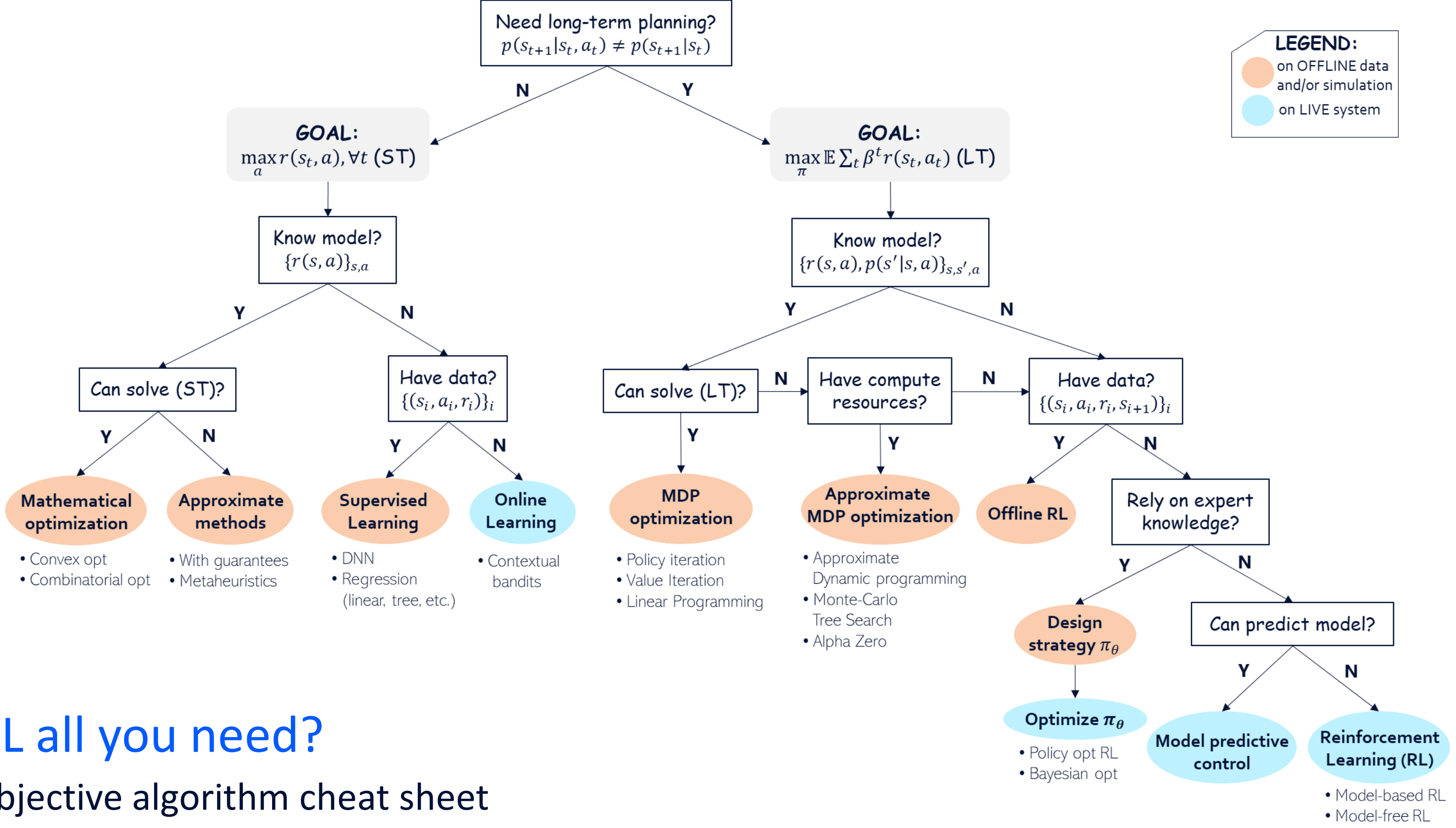
Goal: $\max_{\pi} \mathbb{E} \sum_t \beta^t r(s_t, a_t)$

Is RL all you need?

- MDP formulation is (overly) appealing:
  - ✓ It is general and can describe many real problems!
  - ✓ Can be solved via "standard" Reinforcement Learning (RL)

- Yet:
  - ✓ There exist several sub-variants of the general MDP model
  - ✓ ...with ad-hoc algorithms converging faster than RL!

- *if all you have is a hammer, everything looks like a nail* ☺



Is RL all you need?

A subjective algorithm cheat sheet

Who needs planning?

Need long-term planning?
$p(s_{t+1}|s_t, a_t) \neq p(s_{t+1}|s_t)$

N

GOAL:
$\max_a r(s_t, a), \forall t$ (ST)

Y

GOAL:
$\max_\pi \mathbb{E} \sum_t \beta^t r(s_t, a_t)$ (LT)

E.g., **News recommendation:** Recommend the appropriate news (=action) to the next person (=state)

- ❖ no impact of action on state evolution
- ❖ act greedily, just care for the present!

From "LinUCB" paper

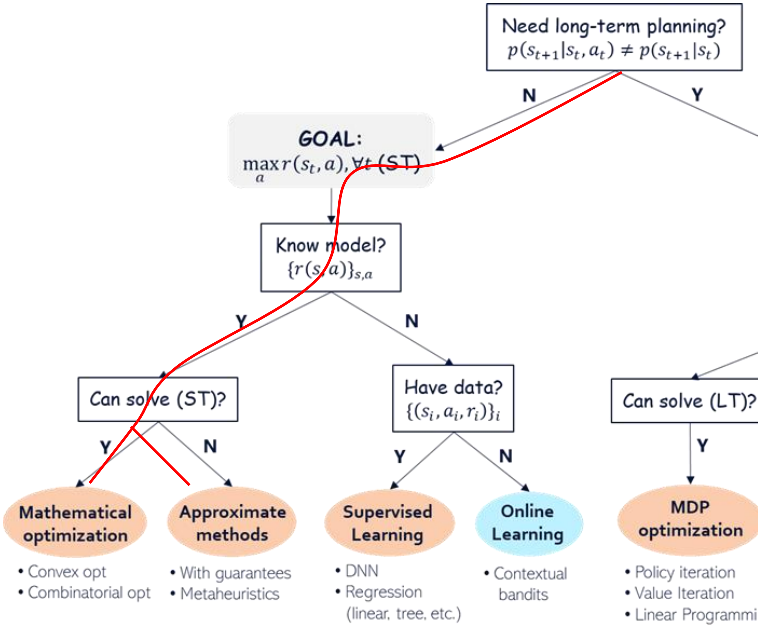
E.g., **Chess:** By moving a piece (=action) the board (=state) changes

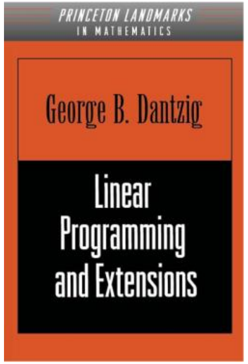
- ❖ the action impacts the state evolution
- ❖ plan ahead!

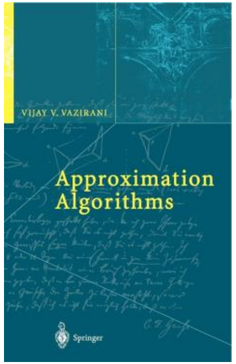
No planning, Model known

Static optimization

- No planning: **Greedily** $\max_a r(s_t, a), \forall t$
- If the reward function r is known, then this boils down to **classic (static) optimization**!







NOKIA

No planning, Model unknown, Have data

Supervised learning

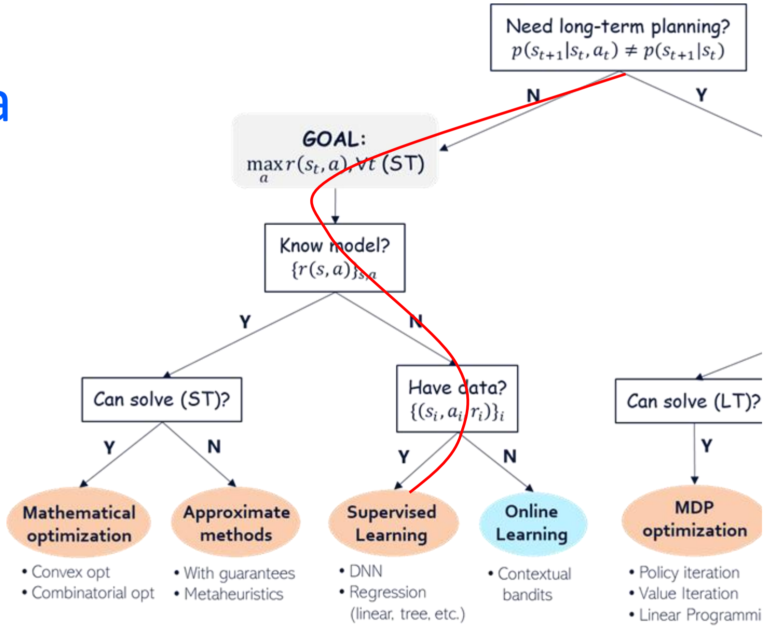
- No planning: Greedily $\max_a r(s_t, a), \forall t$
- The model (reward function r) is unknown
- Yet, we have **historical data** containing tuples:
  - ✓ State s_i
  - ✓ Action a_i
  - ✓ (noisy) Reward r_i

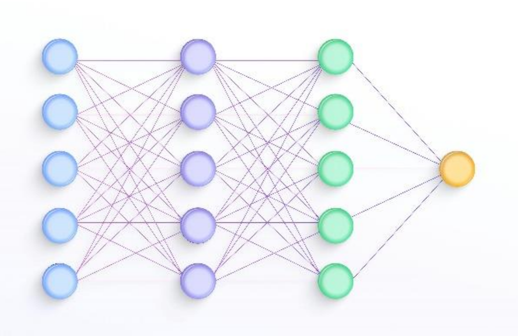
→ We can approximate $\{r(s,a)\}_{s,a}$ via **supervised learning**:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_i (r_\theta(s_i, a_i) - r_i)^2$$

where r_θ can be the output of a neural network with weights & biases θ

→ At each time t, find optimal action by $\max_a r_\theta(s_t, a), \forall t$



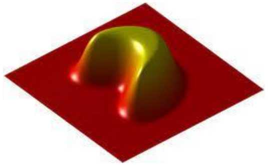


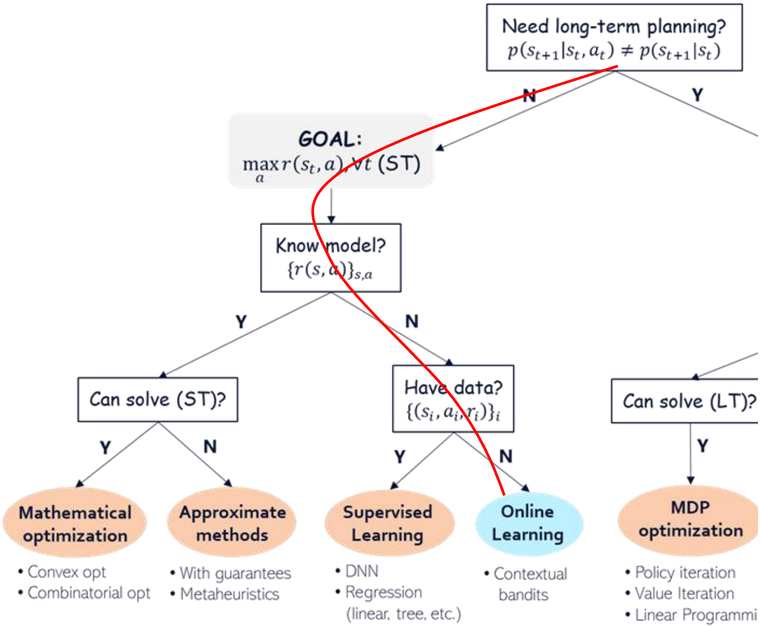
No planning, Model unknown, No data

Online learning

- No planning: Greedily $\max_a r(s_t, a), \forall t$
- The model (reward function r) is unknown
- No historical data

- Optimize r while learning it!
  - ✓ If reward is uncorrelated across states and actions:
    → a different **multi-armed bandit** in each state
  - ✓ If reward is correlated across states but not actions:
    → LinUCB: $r(s,a) \approx \theta_{a,1} s + \theta_{a,2}$
  - ✓ If reward is correlated both across states and actions:
    → Contextual Gaussian Processes

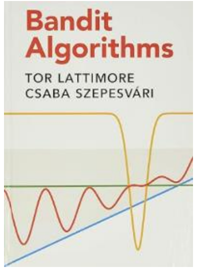




LinUCB


A Contextual-Bandit Approach to Personalized News Article Recommendation

Lihong Li†, Wei Chu†, †Yahoo! Labs, lihong,chuwei@yahoo-inc.com
John Langford‡, ‡Yahoo! Labs, jl@yahoo-inc.com
Robert E. Schapire+*, +Dept of Computer Science, Princeton University, schapire@cs.princeton.edu




Contextual Gaussian Process Bandit Optimization

Andreas Krause, Cheng Soon Ong
Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland
krausea@ethz.ch chengsoon.ong@inf.ethz.ch

Planning, Model known, Easy MDP

Solve an MDP

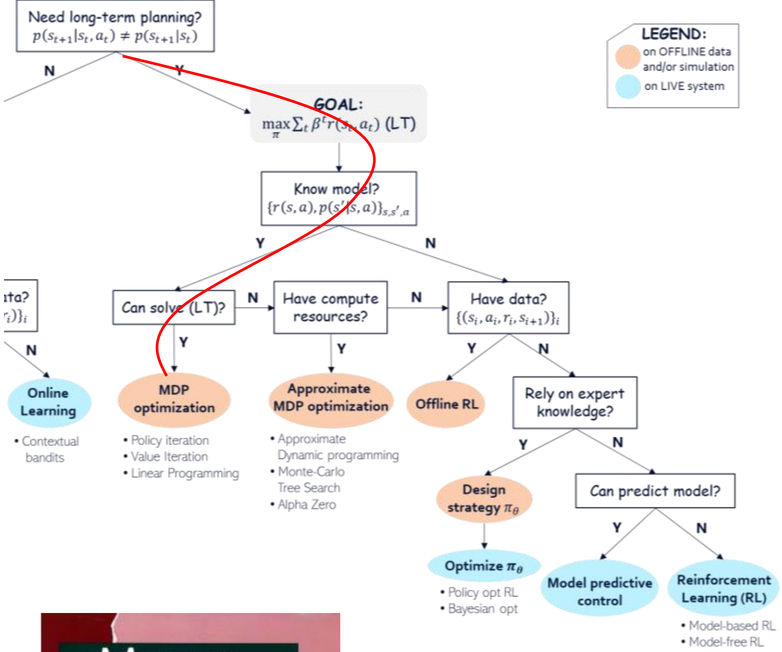
- Need long-term **planning**:

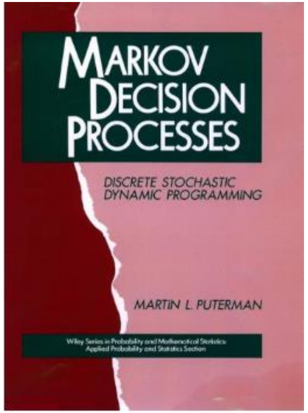
$$\max_{\pi} \mathbb{E} \sum_t \beta^t r(s_t, a_t)$$

- **Lucky** enough to **know the model**:
  - reward function $\{r(s,a)\}_{\forall s,a}$
  - transition probabilities $\{p(s'|s,a)\}_{\forall s,s',a}$

- **Lucky** enough that state/action space is so **small** that the **MDP** is **solvable** via
  - policy / value iteration / linear programming etc.

- Unfortunately, in practice, *this rarely occurs…*

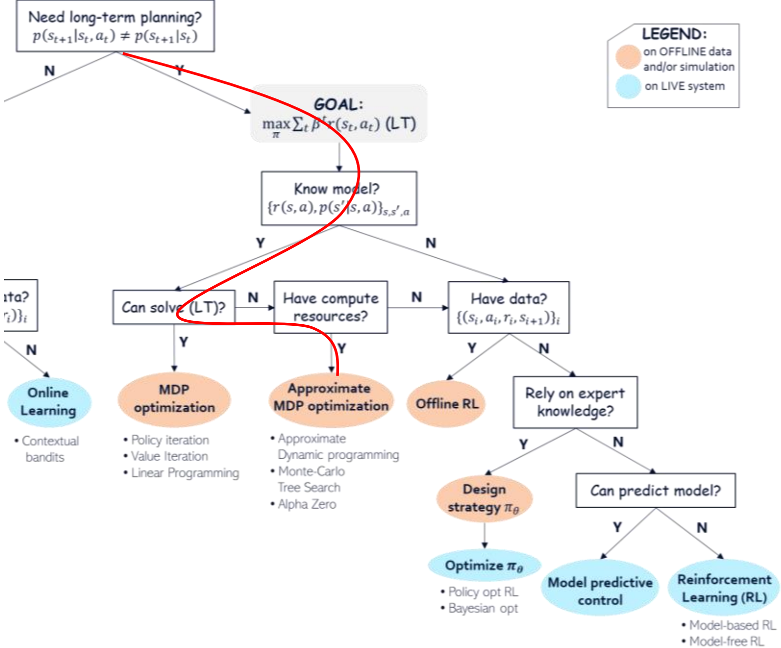


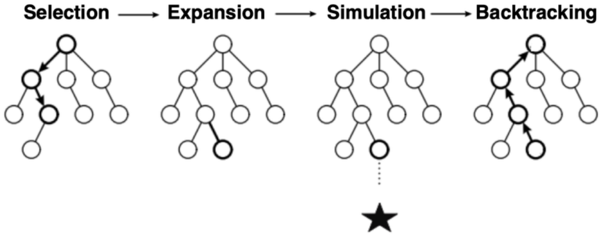


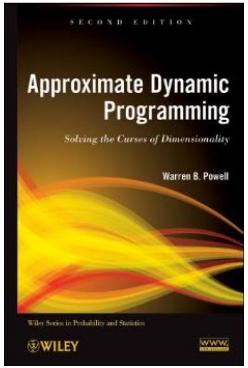
Planning, Model known, Hard MDP

Simulation-based search

- Model is known
- ...but MDP cannot be solved exactly: curse of dimensionality
- Simulate the model and apply:
  - ✓ Approximate Dynamic Programming
  - ✓ Monte-Carlo Tree Search
  - ✓ Alpha Zero







Bandit based Monte-Carlo Planning

Levente Kocsis and Csaba Szepesvári

Computer and Automation Research Institute of the
Hungarian Academy of Sciences, Kende u. 13-17, 1111 Budapest, Hungary
kocsis@sztaki.hu

Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm

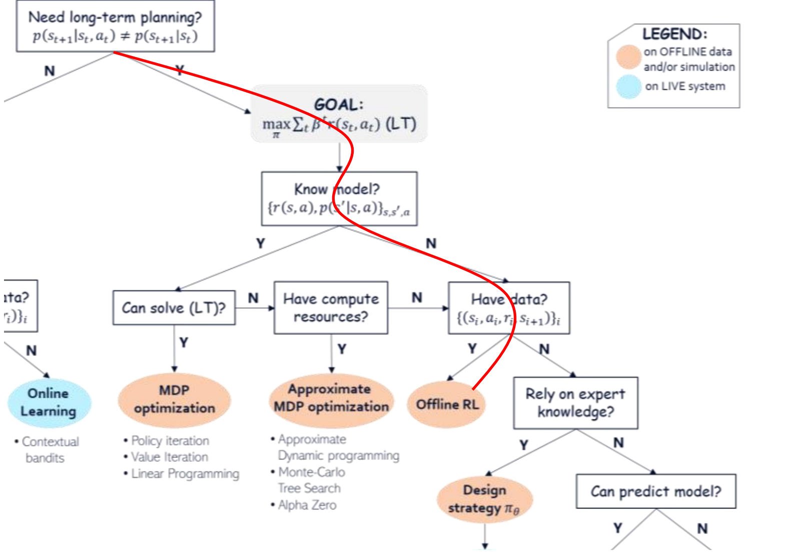
David Silver,[1*] Thomas Hubert,[1*] Julian Schrittwieser,[1*]
Ioannis Antonoglou,[1] Matthew Lai,[1] Arthur Guez,[1] Marc Lanctot,[1]
Laurent Sifre,[1] Dharshan Kumaran,[1] Thore Graepel,[1]
Timothy Lillicrap,[1] Karen Simonyan,[1] Demis Hassabis[1]

[1]DeepMind, 6 Pancras Square, London N1C 4AG.

Planning, Model unknown, Lots of data

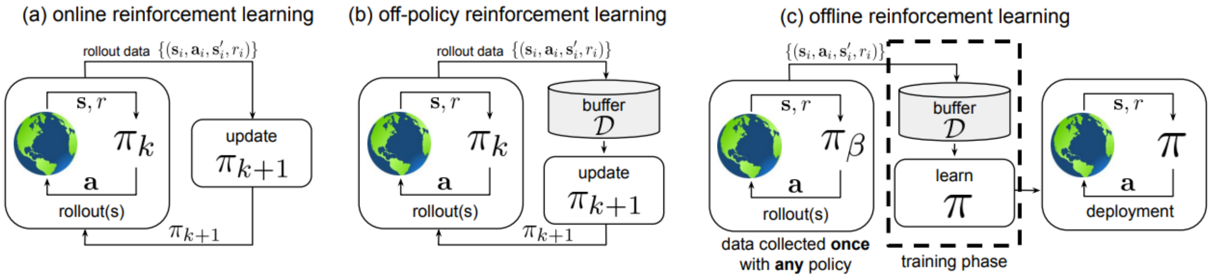
Offline Reinforcement Learning

- Need long-term planning
- Model unknown
- Have (lots of) historical data
- Analogous to supervised learning, but in dynamic settings:
  Offline Reinforcement Learning
- Holds tremendous promise
- Yet, difficult to put into practice:
  counterfactual “what if” queries are impossible!



Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems

Sergey Levine[1,2], Aviral Kumar[1], George Tucker[2], Justin Fu[1]
[1]UC Berkeley, [2]Google Research, Brain Team



Abstract

In this tutorial article, we aim to provide the reader with the conceptual tools needed to get started on research on offline reinforcement learning algorithms: reinforcement learning algorithms that utilize previously collected data, without additional online data collection. Offline reinforcement learning algorithms hold tremendous promise for making it possible to turn large datasets into powerful decision making engines. Effective offline reinforcement learning methods would be able to extract policies with the maximum possible utility out of the available data, thereby allowing automation of a wide range of decision-making domains, from healthcare and education to robotics. However, the limitations of current algorithms make this difficult. We will aim to provide the reader with an understanding of these challenges, particularly in the context of modern deep reinforcement learning methods, and describe some potential solutions that have been explored in recent work to mitigate these challenges, along with recent applications, and a discussion of perspectives on open problems in the field.

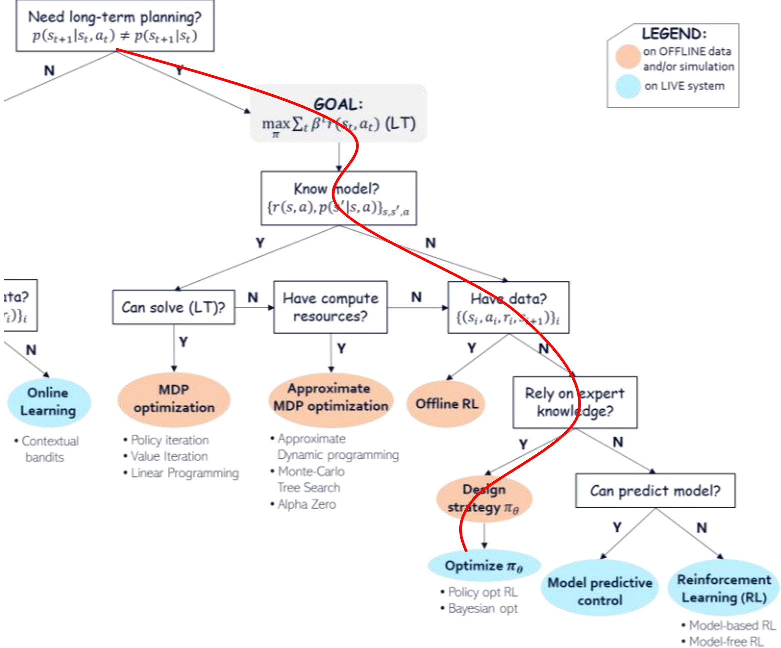
Planning, Rely on expert knowledge

Policy optimization

- Need long-term planning
- Model unknown
- **Domain expert** has **designed** a policy π_θ parametrized by θ
- **Goal:** optimize θ
  - **Option 1: à la RL:** policy gradient, Proximal Policy Optimization (PPO, used for LLMs), etc.

$$\theta \leftarrow \theta + \sum_{t=1}^{T} G_t \nabla_\theta \log \pi_\theta(s_t, a_t)$$

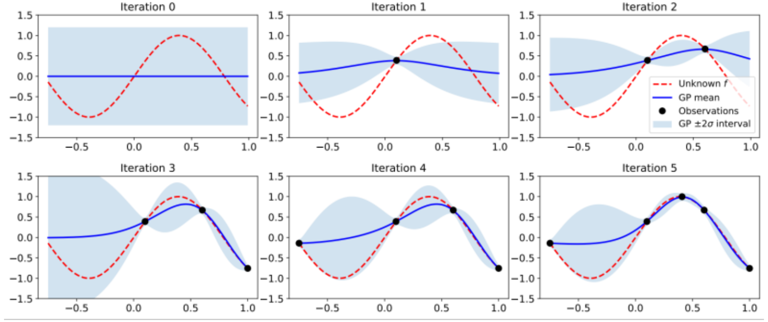
  - **Option 2: Black-box**, e.g., Bayesian **optimization**



Proximal Policy Optimization Algorithms

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
{joschu, filip, prafulla, alec, oleg}@openai.com

Bayesian optimization



Predict & Plan

Model predictive control

- Need long-term planning
- **Model** can be **predicted over a short future time horizon** T

→ **Model predictive control:** At each time step t

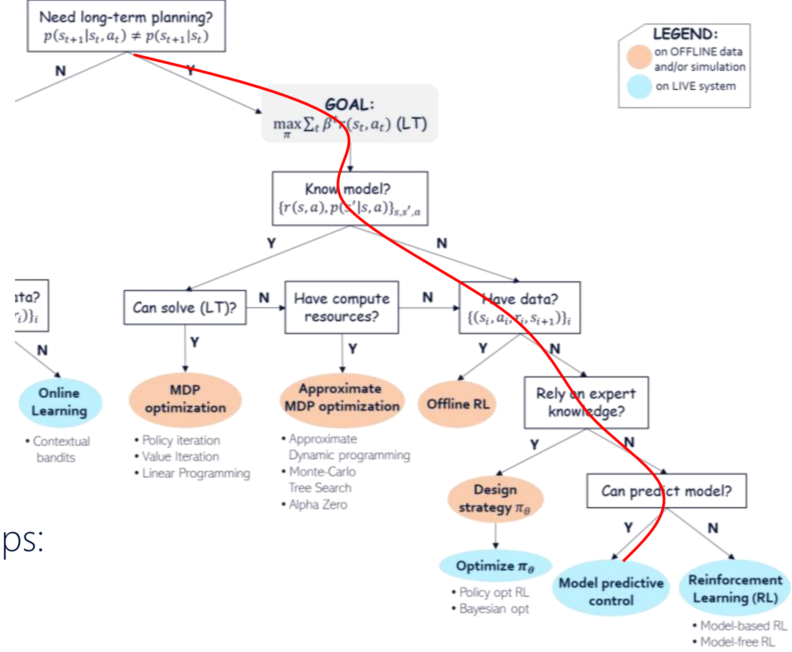
i. **Predict** state transition $\hat{p}$ and reward $\hat{r}$ over next T steps

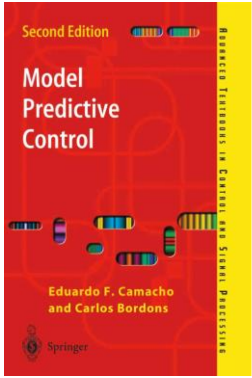
ii. **Compute** the (deterministic) optimal strategy over next T steps:

$$\hat{\pi}_{t+1}, \dots, \hat{\pi}_{t+T} = \arg\max_{\pi} \mathbb{E} \sum_{i=1}^{T} \hat{r}(s_{t+i}, a_{t+i})$$

iii. **Implement** the strategy $\hat{\pi}_{t+1}$ only at next step

- In practice, **works well** even if predictions are poor
- **Successful** industrial applications (chemical plants, oil refineries, power systems)
- Yet, requires high online computational complexity

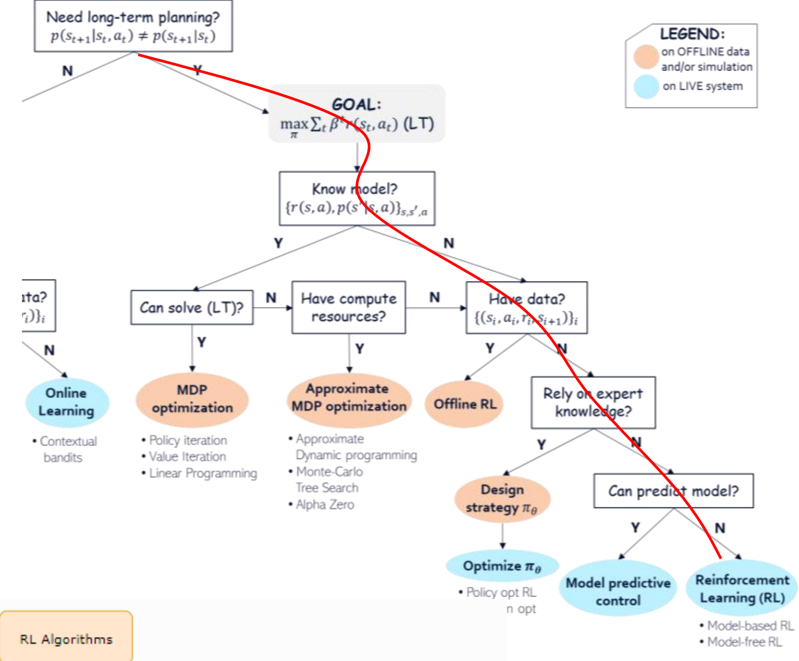


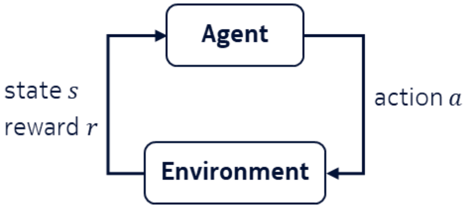


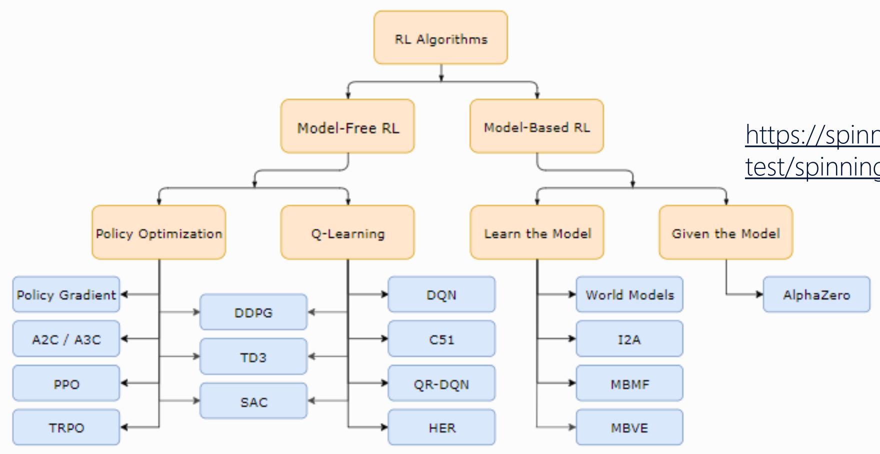
Last but not least….RL!

- Need long-term planning
- Model is unknown and unpredictable
- Expert domain strategy is not good enough

→ "Full-blown" Reinforcement Learning







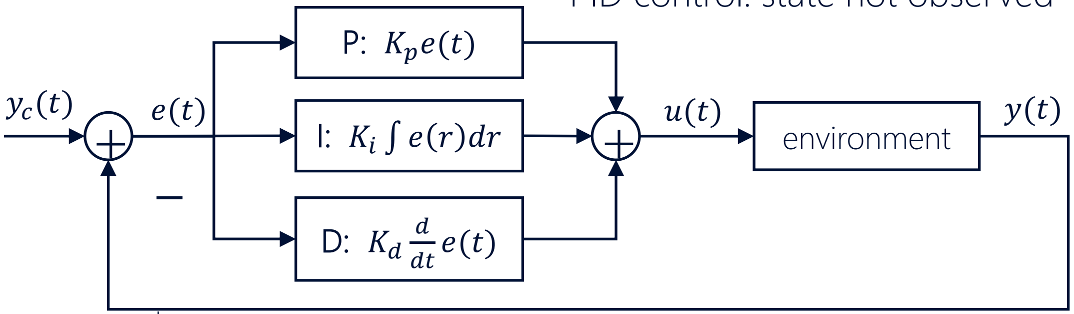
https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html

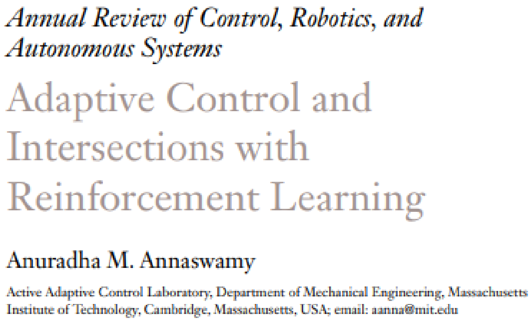
Bonus: PID/Adaptive control

- Action (knob) is unidimensional (e.g., car's wheel steering angle)
- Time is continuous
- When action $a(t)$ is applied, output $y(t)$ is produced (e.g., car position)
- Reference $y_c(t)$ is the desired output (e.g., car in the middle of the lane)
- Reward has the form $e(t) = y(t) - y_c(t)$
- Goal: $\lim_{t \to \infty} e(t) = 0$

→ Proportional-Integral-Differential (PID) or adaptive control

PID control: state not observed





Annual Review of Control, Robotics, and Autonomous Systems

Adaptive Control and Intersections with Reinforcement Learning

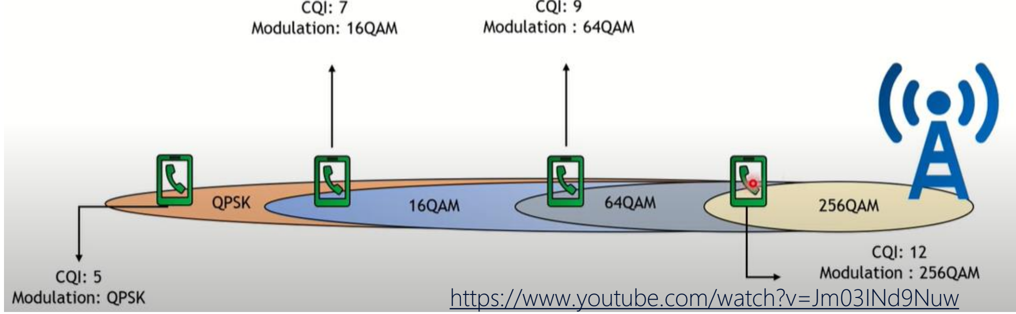
Anuradha M. Annaswamy

Active Adaptive Control Laboratory, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA; email: aanna@mit.edu

Application to Link Adaptation & Scheduling

Link Adaptation (LA)

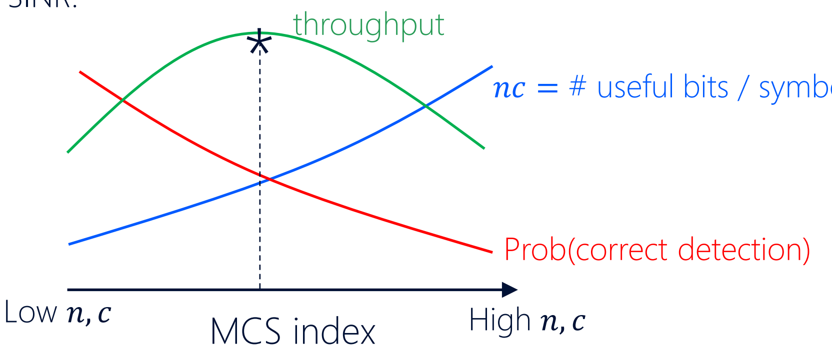
General objective



- Depending on the channel quality (**SINR**), we want to **adapt** Modulation & Coding Scheme (**MCS**)
  - n = # bits / per symbol (**modulation scheme**)
  - c = # bits of information / total # bits—counting redundancy for error correction (**code rate**)

  to **maximize user throughput**

Given the current SINR:

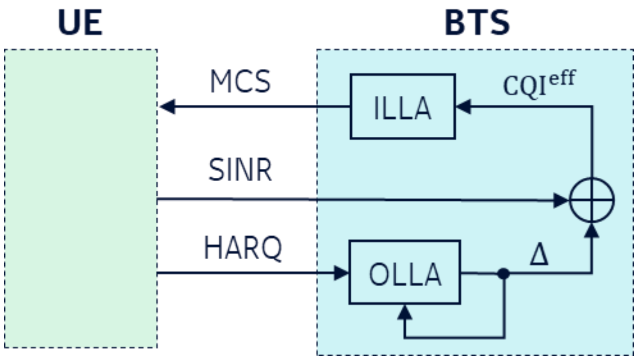


State of the Art

Inner & Outer Loop Link Adaptation (ILLA & OLLA)

eOLLA: an enhanced outer loop link adaptation for cellular networks

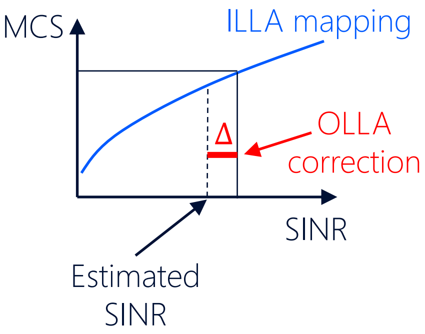
Francisco Blanquez-Casado*, Gerardo Gomez, Maria del Carmen Aguayo-Torres and Jose Tomas Entrambasaguas



- **ILLA:** static mapping SINR → MCS
  - Computed as the MCS guaranteeing BLER=X% (e.g., 10%)
  - *Rational:* BLER not too high (too many retransmissions)
    not too low (too much overhead)
- **Pb:** SINR is poorly estimated by UE

→ **OLLA** provides corrective factor Δ to estimated SINR

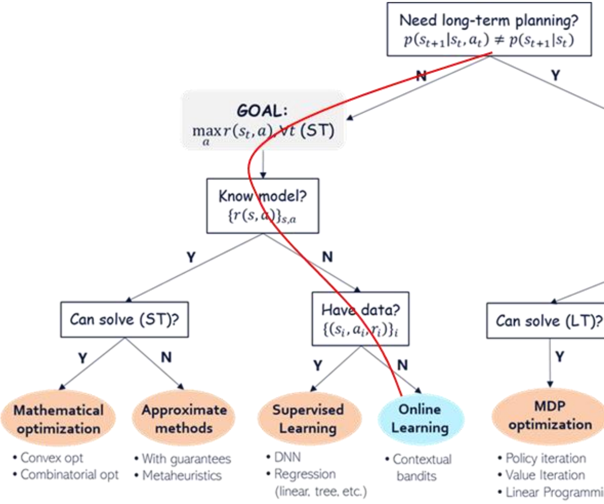
$$\Delta_t = \begin{cases} \Delta_{t-1} + \theta_1 & \text{if } \mathrm{HARQ}_t = \mathrm{ACK} \\ \Delta_{t-1} - \theta_2 & \text{if } \mathrm{HARQ}_t = \mathrm{NACK} \end{cases}$$



1) LA via Online learning

- Set goal: Maintain BLER = $\boldsymbol{X}$%, e.g., $\mathbf{10\%}$
  - ✓ Given current "state" s_t (SINR estimates over recent past)
  - ✓ Find action a_t (MCS)
  - ✓ Such that $\mathbf{Pr(Block\ error\ at\ time}\ t) \coloneqq f(a_t, s_t) = X$
- Reward: $-|f(a_t, s_t) - X|$
- No planning needed: action does not impact state evolution
- Pb: learn f while optimizing it

→ Online learning



Bayesian Link Adaptation under a BLER Target

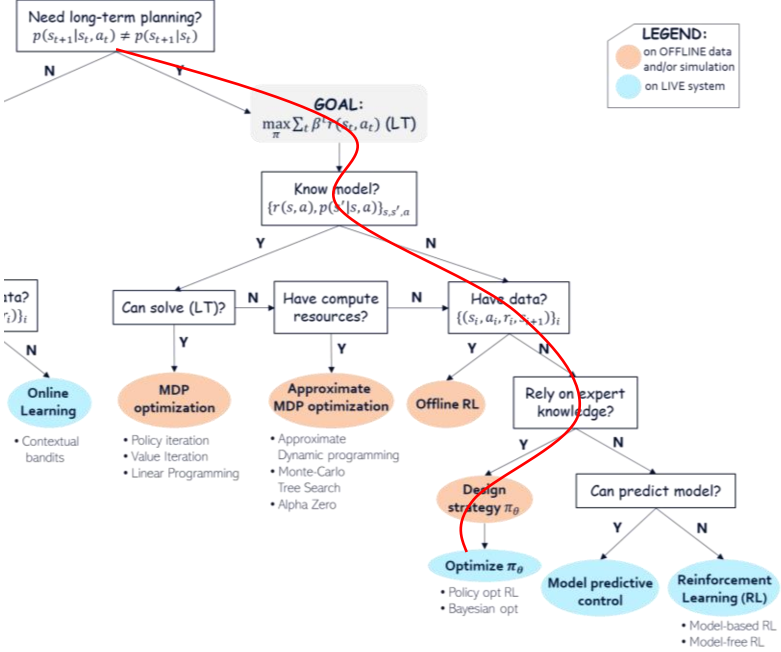
Vidit Saxena*† and Joakim Jaldén*
vidits@kth.se jalden@kth.se
*KTH Royal Institute of Technology, Stockholm, Sweden
†Ericsson Research, Stockholm, Sweden

2) LA via Policy optimization

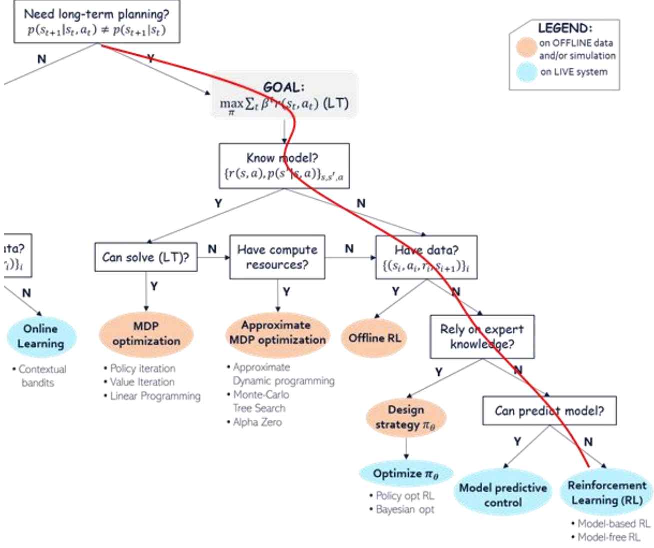
Expert designed

- Define **state** $s_t = (\Delta_{t-1}, \mathrm{SINR}_t, \mathrm{HARQ}_t)$
- **Action** $a_t \in A$ is MCS used for next transmission
- Deterministic ILLA+OLLA (expert designed) **policy**:
  $\pi_\theta(s_t) = \mathrm{ILLA}(\mathrm{SINR}_t + \Delta_t(\Delta_{t-1}, \mathrm{HARQ}_t)), \forall t$
  where $\Delta_t = \begin{cases} \Delta_{t-1} + \theta_1 & \text{if } \mathrm{HARQ}_t = \mathrm{ACK} \\ \Delta_{t-1} - \theta_2 & \text{if } \mathrm{HARQ}_t = \mathrm{NACK} \end{cases}$
- **Reward** = throughput
- Note: **Planning is needed** (state evolution depends on action)

→ Optimize θ via **policy optimization**



3) LA via Reinforcement Learning

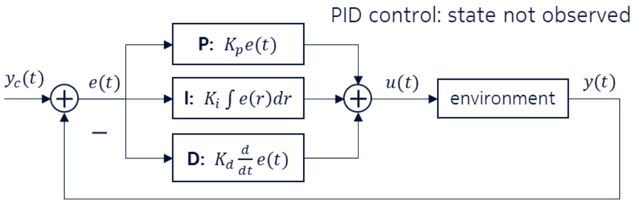


- Define rich **state**:
  - CQI
  - #ACK, #NACK over a recent time window
  - Last MCS
  - Buffer state, etc.
- **Action** = MCS for next transmission or OLLA SINR corrective factor Δ_t
- **Reward** = throughput
- No pre-defined parametrized policy

→ Full-blown Reinforcement Learning

4) LA via Adaptive control

- **Action / Knob:** MCS index
- **Produced output:** BLER
- **Reference:** X%
- **Goal:** Error = BLER – X% = 0



Conclusions

- The **MDP** model is **general** and can be solved via RL

→ Strong **temptation** to use RL everywhere!

- Yet, MDP model boils down to **simpler** models (with ad-hoc algorithms) depending on:
  - whether long-term **planning** is needed
  - how much **data** is available
  - whether a reliable **simulator** is available
  - whether we can rely on **domain-expert** knowledge

Machine Learning without tears

Check out our blog! ☺
https://mlwithouttears.com/

Blog post

MATHY STUFF, HOW I WOULD HAVE LIKED TO LEARN THEM

Fifty (four, actually) shades of conformal prediction
February 4, 2024

In this post we review different methods to compute prediction intervals, containing the next (unknown) observation with high probability and being at the heart of Conformal Prediction (CP). We will highlight that each method is characterized by a different and non-trivial trade-off between computational complexity, coverage properties and the size of the prediction interval. Scenario. We are...

Conformal prediction

Conformalized quantile regression
January 17, 2024

Pimp quantile regression with strong coverage guarantees Suppose that we are given a historical dataset containing samples of the form , where and are the -th realizations of (predictor) variable and of (predicted) variable , respectively. As a running example, let us consider the following dataset: Our goal #1 is to estimate the trend of variable...

Conformal prediction

Quantile regression
January 3, 2024

An expressive and robust alternative to least square For regression problems, least square regression (LSR) arguably gets the lion share of data scientists' attention. The reasons are several: LSR is taught in virtually every introductory statistics course, it is intuitive and is readily available in most of software libraries. LSR estimates the mean of the predicted variable...

NOKIA