# A Quick Look at the "Reinforcement Learning" course

A. LAZARIC (*SequeL Team @INRIA-Lille*)
*Ecole Centrale - Option DAD*

SequeL – INRIA Lille

# Why

# Why: Important Problems

# Why: Important Problems

- Autonomous robotics

# Why: Important Problems

- Autonomous robotics
- *Elder care*

# Why: Important Problems

- Autonomous robotics

- *Elder care*
- *Exploration of unknown/dangerous environments*

# Why: Important Problems

- Autonomous robotics

- *Elder care*
- *Exploration of unknown/dangerous environments*
- *Robotics for entertainment*

# Why: Important Problems

- Autonomous robotics
- Financial applications

# Why: Important Problems

- Autonomous robotics
- Financial applications

- *Trading execution algorithms*

# Why: Important Problems

- Autonomous robotics
- Financial applications

- *Trading execution algorithms*
- *Portfolio management*

# Why: Important Problems

- Autonomous robotics
- Financial applications

- *Trading execution algorithms*
- *Portfolio management*
- *Option pricing*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management

- *Energy grid integration*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management

- *Energy grid integration*
- *Maintenance scheduling*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management

- *Energy grid integration*
- *Maintenance scheduling*
- *Energy market regulation*

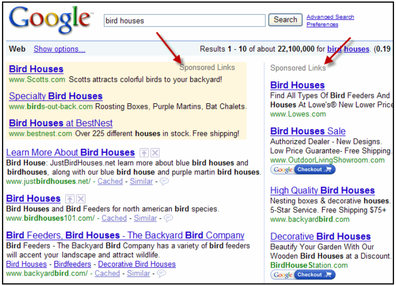
# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management

- *Energy grid integration*
- *Maintenance scheduling*
- *Energy market regulation*
- *Energy production management*

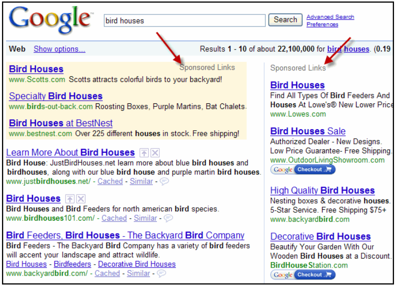
# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems



# Why: Important Problems

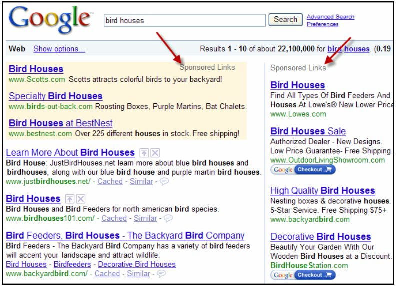
- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems




- *Web advertising*

# Why: Important Problems

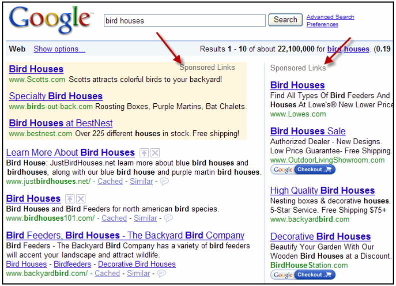
- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems




- *Web advertising*
- *Product recommendation*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems




- *Web advertising*
- *Product recommendation*
- *Date matching*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems
- Social applications

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems
- Social applications


- *Bike sharing optimization*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems
- Social applications


- *Bike sharing optimization*
- *Election campaign*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems
- Social applications


- *Bike sharing optimization*
- *Election campaign*
- *ER service optimization*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems
- Social applications

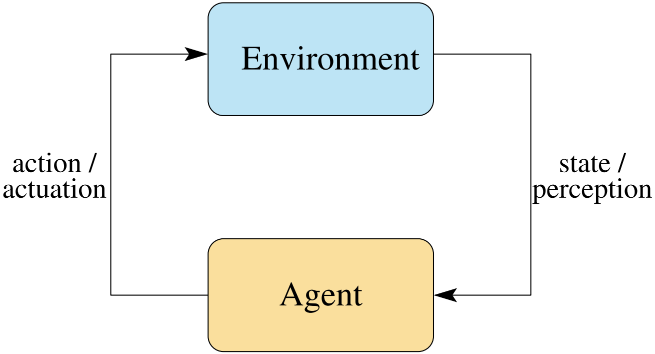

- *Bike sharing optimization*
- *Election campaign*
- *ER service optimization*
- *Resource distribution optimization*

# Why: Important Problems

- Autonomous robotics
- Financial applications
- Energy management
- Recommender systems
- Social applications
- And many more...

# What

# What: Decision-Making under Uncertainty



# How: Reinforcement Learning

*__Reinforcement learning__ is learning what to do – how to map situations to actions – so as to maximize a numerical reward signal. The learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them (__trial–and–error__). In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards (__delayed reward__).*

*"An introduction to reinforcement learning",*
*Sutton and Barto (1998).*

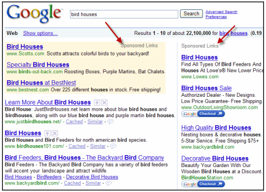
# How: the Course

# How: the Course

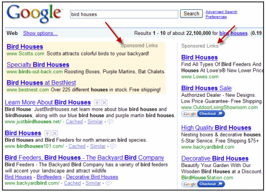
# How: the Course

# How: the Course

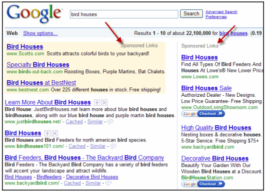
# How: the Course



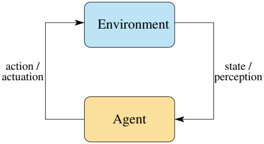
# How: the Course



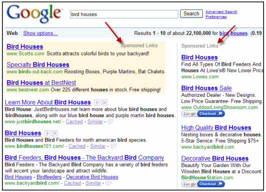

# How: the Course



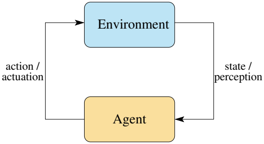




# How: the Course








*Formal* and *rigorous* approach to
the RL's way to decision-making under uncertainty

# What: the Highlights of the Course

**How do we formalize the agent-environment interaction?**

# What: the Highlights of the Course

**How do we formalize the agent-environment interaction?**

## Markov Decision Process and Policy

A Markov decision process (MDP) is represented by the tuple $M = \langle X, A, r, p\rangle$ where $X$ is the state space, $A$ is the action space, $r : X \times A \to [0, B]$ is the reward function, $p$ is the dynamics.
At time $t \in \mathbb{N}$ a *decision rule* $\pi_t : X \to A$ is a mapping from states to actions and a *policy* (strategy, plan) is a sequence of decision rules $\pi = (\pi_0, \pi_1, \pi_2, \dots)$.

## The Bellman equations

$$V^\pi(x) = r(x, \pi(x)) + \gamma \sum_y p(y|x, \pi(x)) V^\pi(y),$$

$$V^*(x) = \max_{a \in A} \Big[ r(x, a) + \gamma \sum_y p(y|x, a) V^*(y) \Big].$$

# What: the Highlights of the Course

**How do we solve an MDP?**

# What: the Highlights of the Course

**How do we solve an MDP?**

## Dynamic Programming

*Value Iteration*

$$V_{k+1} = \mathcal{T} V_k$$

*Policy Iteration*

- *Evaluate*: given $\pi_k$ compute $V^{\pi_k}$.
- *Improve*: given $V^{\pi_k}$ compute $\pi_{k+1} = \text{greedy}(V^{\pi_k})$

# What: the Highlights of the Course

**How do we solve an MDP "online"?**

# What: the Highlights of the Course

**How do we solve an MDP "online"?**

## Q-learning

Given a observed transition $x, a, x', r$ update

$$Q_{k+1}(x,a) = (1-\alpha)Q_k(x,a) + \alpha\big(r + \max_{a'} Q_k(x',a')\big).$$

# What: the Highlights of the Course

**How do we effectively trade-off exploration and exploitation?**

# What: the Highlights of the Course

**How do we effectively trade-off exploration and exploitation?**

## Multi-arm Bandit

Given $K$ arms we define the regret over $n$ rounds of a bandit strategy as

$$R_n = \sum_{t=1}^{n} X_{i^*,t} - \sum_{t=1}^{n} X_{I_t,t}.$$

For the UCB strategy we can prove

$$R_n \leq \sum_{i \neq i^*} \frac{b^2}{\Delta_i} \log(n).$$

# What: the Highlights of the Course

**How do we solve a "huge" MDP?**

# What: the Highlights of the Course

**How do we solve a "huge" MDP?**

## Approximate Dynamic Programming

*Approximate* Value Iteration

$$\hat{V}_{k+1} = \widehat{\mathcal{T}}\hat{V}_k$$

*Approximate* Policy Iteration

- *Evaluate*: given $\pi_k$ compute $\hat{V}^{\pi_k}$.
- *Improve*: given $\hat{V}^{\pi_k}$ compute $\hat{\pi}_{k+1} \approx \text{greedy}(\hat{V}^{\pi_k})$

# What: the Highlights of the Course

**How “sample-efficient” are these algorithms?**

# What: the Highlights of the Course

**How "sample-efficient" are these algorithms?**

## Sample Complexity of LSPI

$$||V^{\pi_K} - V^*||_{2,\rho} \leq \inf_{f \in \mathcal{F}} ||V^* - f||_{2,\rho} + \frac{C_\rho}{1-\gamma}\sqrt{\frac{\log(1/\delta)}{n}}.$$

# Who

*Lectures and Practical Sessions*

*Alessandro LAZARIC*

SequeL Team
INRIA-Lille Nord Europe

alessandro.lazaric@inria.fr
researchers.lille.inria.fr/~lazaric/

# When/What/Where

*See planning on the website.*

# Evaluation

- To be defined (probably homework+review project at the end of the course)

# Reinforcement Learning

*Alessandro Lazaric*
alessandro.lazaric@inria.fr
sequel.lille.inria.fr