Lower bounds in reinforcement learning: the intelligent agent dream is getting closer

D. Ernst
University of Liège

TU Delft - The 13th of January 2009

Reinforcement learning for artificial intelligent agents

An artificial intelligent agent is anything we create that is capable of taking actions based on information it perceives, on its own experience, and on its own decisions about which actions to perform.

Reinforcement Learning (RL) refers to a class of problems which postulate an artificial intelligent agent exploring an environment in which the agent perceives information about its current state and takes actions. The environment, in return, provides a reward signal (which can be positive or negative). The agent has as objective to maximize the cumulative reward signal over the course of the interaction.

Environment

with internal state *x*

Reward signal *r*

Observation signal *s*

Action *u*

The reinforcement learning agent

Some generic difficulties in RL

Inference problem. The environment dynamics and the mechanism behind the reward signal are (partially) unknown. “Good policies” need to be inferred from the information the agent has gathered from interaction with the system.

Computational complexity. The information has to be processed within limited computing times and memory.

Exploration-exploitation tradeoff. To obtain a lot of reward, a reinforcement learning agent must prefer actions that it has tried in the past and found to be effective in producing reward. But to discover such actions, it has to try actions that it has not selected before.

Inference of near-optimal policies from a set of trajectories

Two main types of approaches:

- Identification of an analytical (or algorithmic) model and derivation from it of an optimal policy. Approach favored by the control community.
- Computation of the optimal policies directly from the trajectories, without relying on the identification of an analytical model. Approach favored by the computer science community.

The direct approach

Classical strategy: to infer from trajectories state-action value functions.

Discrete (and not too large) state and action spaces: state-action value functions can be represented in tabular form.

Continuous or large state and action spaces: function approximators need to be used. Also, learning must be done from finite and generally very sparse sets of trajectories.

Parametric function approximators with gradient descent like methods: very popular techniques but do not generalize well enough to move from the academic to the real-world.

Supervised learning and RL

Problem of generalization over an information space: occurs also in supervised learning (SL).

What is SL ? To infer from a sample of input-output pairs input = information state; output = class label or real number a model which explains “at best” these pairs.

Supervised learning highly successful: state-of-the art SL algorithms have been successfully applied to problems where information state = thousands of components.

Recent RL algorithms: use the generalization capabilities of supervised learning algorithms by solving a sequence of standard supervised learning problems. Most of them are instances of the generic “fitted Q iteration” algorithm, a batch mode reinforcement learning algorithm.

Learning from a sample of trajectories: a typical problem statement

Discrete-time dynamics: $x_{t+1} = f(x_t, u_t) \quad t = 0, 1, \ldots, T-1$
where $x_t \in X$ and $u_t \in U$.
Reward function: $\rho(x, u) : X \times U \to \mathbb{R}$.
Instantaneous reward: $r_t = \rho(x_t, u_t)$.
Type of policies considered: $h : \{0, 1, \ldots, T-1\} \times X \to U$.
Return of a policy: $J^h(x) = \sum_{t=0}^{T-1} \rho(x_t, h(t, x_t))$ with $x_0 = x$.
Optimal policy h^*: Policy that maximizes J^h for all x.

Problem: To find a "good" approximation of h^*.
We do not know: $f(x, u)$ and $\rho(x, u)$.
We know instead: A set of one-step system transitions $\mathcal{F} = \{(x^l, u^l, r^l, y^l)\}_{l=1}^{|\mathcal{F}|}$ where y^l is the state reached after taking action u^l in state x^l and r^l the instantaneous reward associated with the transition.

State-action value functions as indirect vehicles for computing h^*

State-action value functions $Q_N^{h^*} : X \times U \to \mathbb{R}$**:**
$Q_N^{h^*}(x,u) = \rho(x,u) + \sum_{t=T-N+1}^{T-1} \rho(x_t, h^*(t,x_t))$ where
$x_{T-N+1} = f(x,u)$.
Interpretation: $Q_N^{h^*}(x,u)$ gives the sum of rewards from $t = T-N$ to $T-1$ when (i) the system is in state x at $t = T-N$, (ii) action chosen at $t = T-N$ is u and (iii) actions selected afterwards according to h^*.
Iterative computation of the functions $Q_N^{h^*}$**:**
$Q_N^{h^*}(x,u) = \rho(x,u) + \max_{u' \in U} Q_{N-1}^{h^*}(f(x,u),u'), \quad \forall N > 0$ with
$Q_0^{h^*}(x,u) \equiv 0$.
Optimality condition: $h^*(T-N,x) \in \arg\max_{u \in U} Q_N^{h^*}(x,u)$

Fitted Q iteration: the algorithm

Input: The set of one-step system transitions $\mathcal{F} = \{(x^l, u^l, r^l, y^l)\}_{l=1}^{|\mathcal{F}|}$.

The algorithm:

- Iterative computation of (hopefully good) approximations of the functions $Q_1^{h^*}$, $Q_2^{h^*}$, ..., $Q_N^{h^*}$ by solving T standard supervised learning problems.
- The training sample for the N^{th} ($N > 0$) problem is $\left\{ \left((x^l, u^l),\ r^l + \max_{u \in U} \hat{Q}_{N-1}^{h^*}(y^l, u) \right) \right\}_{l=1}^{|\mathcal{F}|}$ with $\hat{Q}_0^{h^*}(x, u) \equiv 0$. From the N^{th} training sample, the supervised learning algorithm outputs $\hat{Q}_N^{h^*}$.

Output: The policy $\hat{h}^*(T - N, x) \in \arg\max_{u \in U} \hat{Q}_N^{h^*}(x, u)$

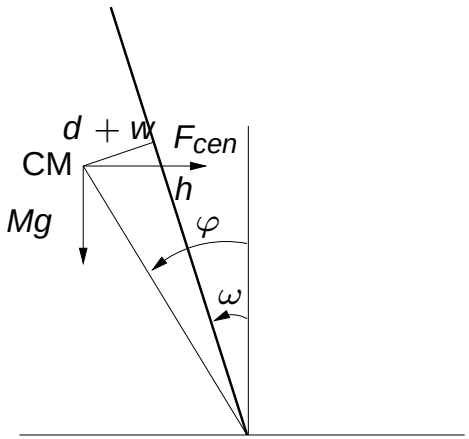
Fitted Q iteration: some remarks

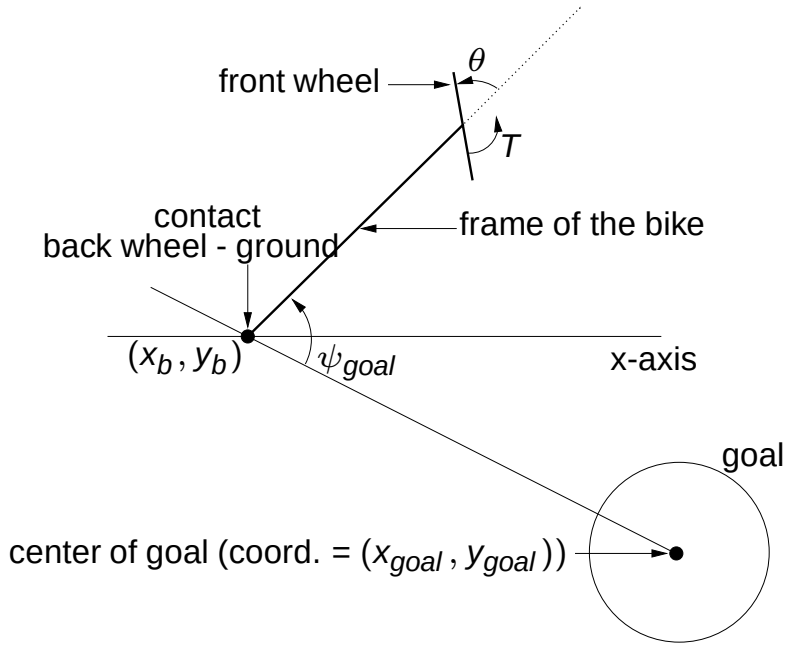
About the performances: Depend on the supervised learning method – Second to none (in terms of sample efficiency) when combined with ensemble of regression trees – Significant computational requirements - Problems when the action space grows.

Extensions: Can be extended to a large class of stochastic optimal control problems.

Theoretical properties: Computed policy converges to the optimal policy under appropriate assumptions on the SL method, the sampling process, the system dynamics and the reward function (typically, Lipschitz continuity).

Bicycle balancing and riding

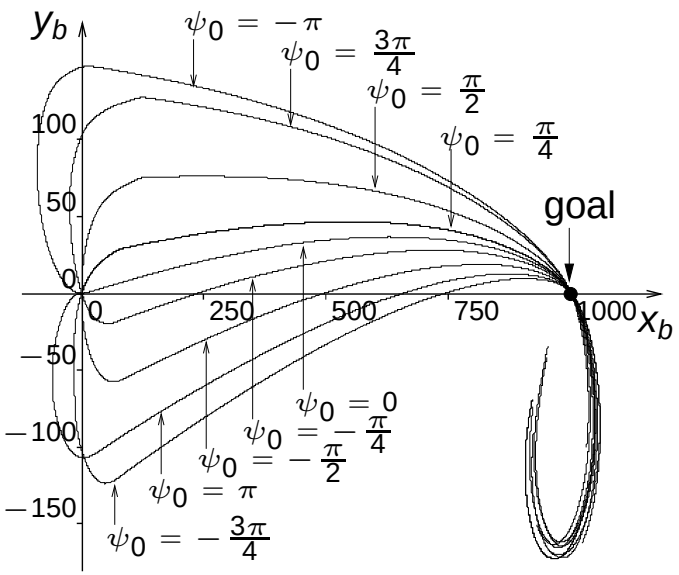


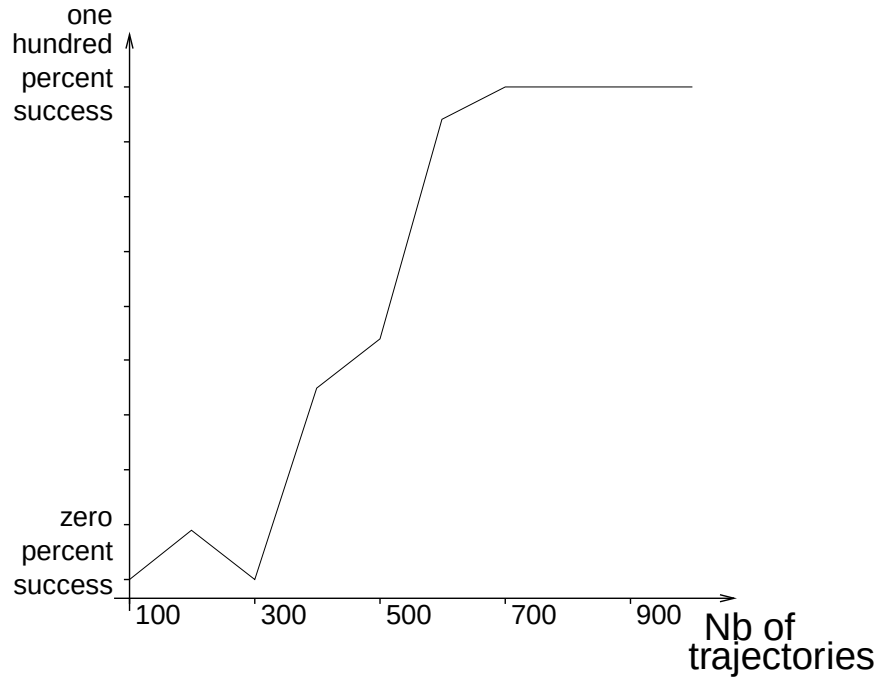


Stochastic problem – Seven state variables and two control actions – Time between t and $t + 1 = 0.01\,s$ – Long optimisation horizon – Reward function of the type:

$$r(x_t, u_t) = \begin{cases} -1 & \text{if bicycle has fallen} \\ -coeff. * (|\psi goal_{t+1}| - |\psi goal_t|) & \text{otherwise} \end{cases}$$

Trajectories generation: actions taken at random, bicycle initially far from the goal and trajectory ends when bicycle falls.



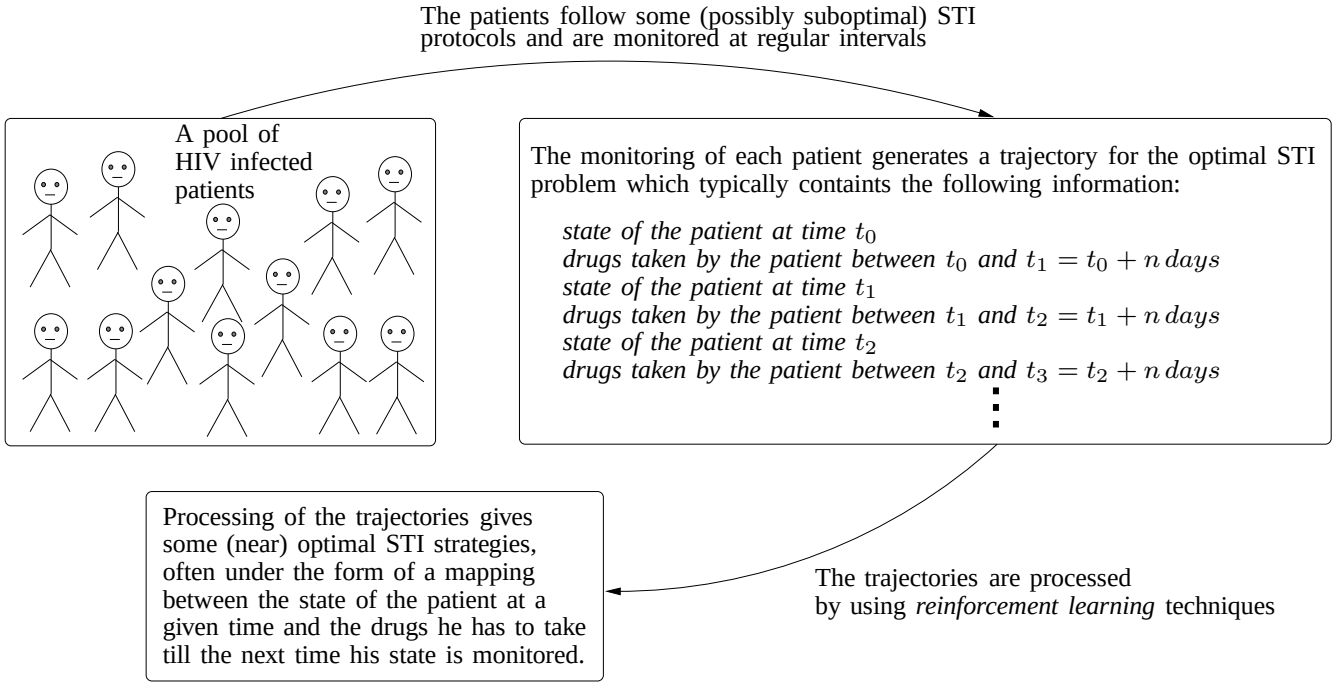


***Q*-learning with neural networks:** 100 times more trajectories are needed.

Computation of Structured Treatment Interruption (STI) for HIV+ patients

STI for HIV: to cycle the patient on and off drug therapy
– In some remarkable cases, STI strategies have enabled the patients to maintain immune control over the virus in the absence of treatment.

RL for designing STI strategies: RL techniques have the potential to infer from clinical data good STI strategies, without modeling the HIV infection dynamics – Clinical data: time evolution of patient's state ($CD4^+$ T cell count, viral load, etc) recorded at discrete-time instants and sequence of drugs administered.



Treatments for many chronic-like diseases could be designed with a similar approach – An active research community in statistics (Google **Dynamic treatment regimes**) focuses on these problems.

Research directions suggested by these two types of applications

Selection of a concise set of system transitions. The size of the different SL learning problems grows with the size of $\mathcal{F}$. After a certain time of interaction, these samples may become so numerous that batch mode RL algorithms become computationally impractical. Selecting a concise set of sufficiently rich representatives of the one-step system transitions may reduce the computational burdens.

Performance guarantees on the policy inferred. It is important to have some guarantees on the quality of the control policies inferred from the one-step system transitions.

Inferring a lower bound on the performance of a control policy from one-step system transitions

Discrete-time dynamics: $x_{t+1} = f(x_t, u_t) \quad t = 0, 1, \ldots, T-1$
where $x_t \in X$ and $u_t \in U$.
Reward function: $\rho(x, u) : X \times U \rightarrow \mathbb{R}$.
Instantaneous reward: $r_t = \rho(x_t, u_t)$.
Type of policies considered: $h : \{0, 1, \ldots, T-1\} \times X \rightarrow U$.
Return of a policy: $J^h(x) = \sum_{t=0}^{T-1} \rho(x_t, h(t, x_t))$ with $x_0 = x$.

Problem: Find a (tight) lower bound on $J^h(x)$.
We do not know: $f(x, u)$ and $\rho(x, u)$.
We know instead: A set of one-step system transitions $\mathcal{F} = \{(x^l, u^l, r^l, y^l)\}_{l=1}^{|\mathcal{F}|}$ where y^l is the state reached after taking action u^l in state x^l and r^l the instantaneous reward associated with the transition.

(Additional assumptions)

The dynamics f, the reward function ρ and the policy h are Lipschitz continuous, i.e., there exist finite constants $L_f, L_\rho, L_h \in \mathbb{R}$ such that:

$$\begin{aligned} \|f(x,u) - f(x',u')\| &\leq L_f\big(\|x - x'\| + \|u - u'\|\big), \quad (1) \\ |\rho(x,u) - \rho(x',u')| &\leq L_\rho\big(\|x - x'\| + \|u - u'\|\big), \quad (2) \\ \|h(t,x) - h(t,x')\| &\leq L_h\|x - x'\|, \quad (3) \end{aligned}$$

$\forall x, x' \in X, \forall u, u' \in U, \forall t \in \{0, \ldots, T-1\}$.

Let us suppose also that three constants L_f, L_ρ, L_h satisfying the above-written inequalities are known.

Computing a lower bound on $J^h(x)$ from a sequence of one-step system transitions

Theorem

Let x be an initial state of the system, h a policy, $\tau = [(x^{l_t}, u^{l_t}, r^{l_t}, y^{l_t})]_{t=0}^{T-1}$ a sequence of one-step system transitions. Then we have the following lower bound:

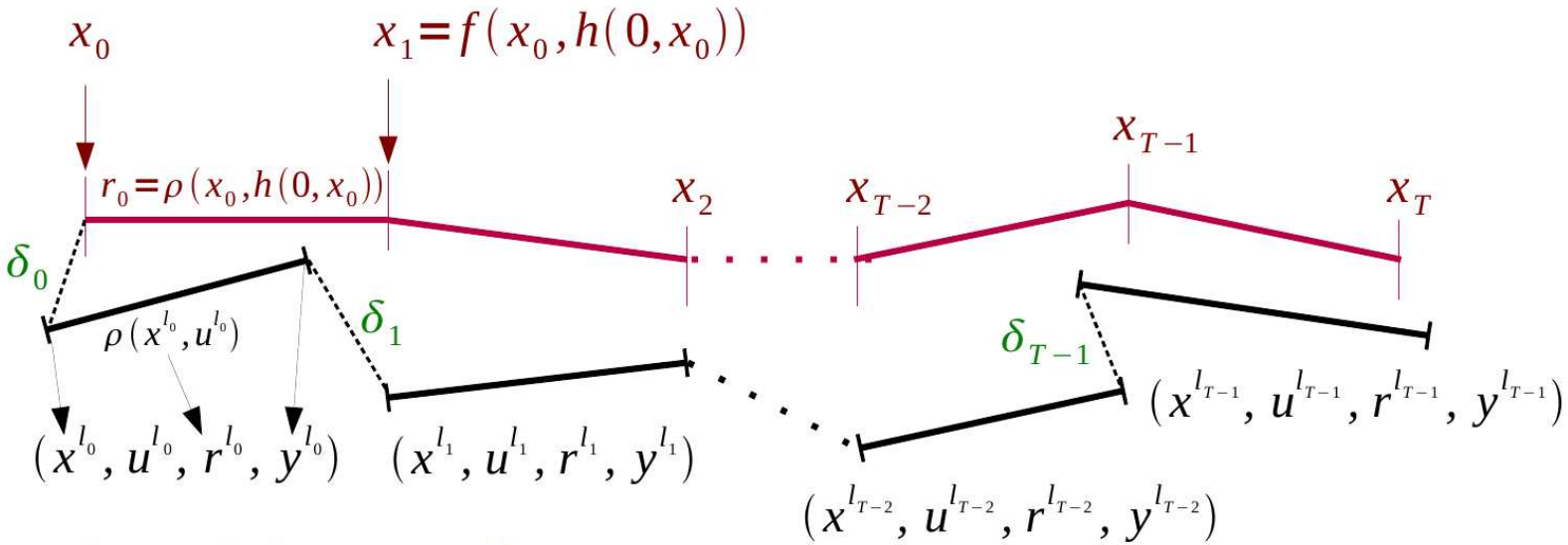
$$\sum_{t=0}^{T-1}(r^{l_t} - L_{Q_{T-t}}\delta_t) \leq J^h(x),$$

where

$\delta_t = \|x^{l_t} - y^{l_{t-1}}\| + \|u^{l_t} - h(t, y^{l_{t-1}})\| \ \forall t \in \{0, 1, \ldots, T-1\}$,

and $L_{Q_N} = L_\rho\left(\sum_{t=0}^{N-1}[L_f(1 + L_h)]^t\right)$ *with* $y^{l_{-1}} = x$.

A graphical interpretation



$$\delta_0 = \left\| x^{l_0} - x_0 \right\| + \left\| u^{l_0} - h(0, x_0) \right\|$$

$$\delta_1 = \left\| y^{l_0} - x^{l_1} \right\| + \left\| u_t^{l_1} - h(1, y^{l_0}) \right\|$$

Finding the highest lower bound

Previously: An approach for computing from a T-length sequence of one-step system transitions a lower bound on $J^h(x)$.

Problem: How to find in a computationally efficient way the sequence of one-step system transitions leading to the highest lower bound (denoted by $B^*_{\mathcal{F}^T}(x)$ later)?

- Naive approach: Exhaustive search over all the elements of $\mathcal{F}^T$.
- Smart approach: See the problem as a problem of finding the shortest path in a graph $-$ Can be solved using a Viterbi-like algorithm $-$ Complexity linear with T and quadratric with $|\mathcal{F}|$.

Tightness of the highest lower bound

Previously: (i) An approach for computing from a T-length sequence of one-step system transitions a lower bound on $J^h(x)$ (ii) An efficient algorithm for computing the T-length sequence of elements of $\mathcal{F}$ leading to the highest lower bound ($B^*_{\mathcal{F}^T}(x)$).

Theorem

Let x be an initial state, h a policy, and $\mathcal{F} = \{(x^l, u^l, r^l, y^l)\}_{l=1}^{|\mathcal{F}|}$ a set of four-tuples. We suppose that $\exists\ \alpha \in \mathbb{R}^+$:

$$\sup_{(x,u)\in X\times U} \Big\{ \min_{l\in\{1,\ldots,|\mathcal{F}|\}} \{\|x^l - x\| + \|u^l - u\|\} \Big\} \leq \alpha, \qquad (4)$$

and we note α^* the smallest constant which satisfies (4). Then

$$\exists\ C \in \mathbb{R}^+ : J^h(x) - B^*_{\mathcal{F}^T}(x) \leq C\alpha^*.$$

Ongoing work and new avenues

Ongoing work:

- Finer delineation of required assumptions.
- Extension to stochastic systems.

New avenues:

- Could serve as the base for designing new RL algorithms which would output policies that lead to the maximisation of these lower bounds.
- Could be used in combination with batch-mode reinforcement learning algorithms for identifying concise sets of one-step system transitions.