# Policy Search in a Space of Simple Closed-form Formulas: Towards Interpretability of Reinforcement Learning

Francis Maes, Raphael Fonteneau, Louis Wehenkel, and Damien Ernst

Department of Electrical Engineering and Computer Science, University of Liège, BELGIUM
{francis.maes, raphael.fonteneau, l.wehenkel, dernst}@ulg.ac.be

**Abstract.** In this paper, we address the problem of computing interpretable solutions to reinforcement learning (RL) problems. To this end, we propose a search algorithm over a space of simple closed-form formulas that are used to rank actions. We formalize the search for a high-performance policy as a multi-armed bandit problem where each arm corresponds to a candidate policy canonically represented by its shortest formula-based representation. Experiments, conducted on standard benchmarks, show that this approach manages to determine both efficient and interpretable solutions.

**Keywords:** Reinforcement Learning, Formula Discovery, Interpretability

## 1 Introduction

Reinforcement learning refers to a large class of techniques that favor a sampling-based approach for solving optimal sequential decision making problems. Over the years, researchers in this field have developed many efficient algorithms, some of them coming with strong theoretical guarantees, and have sought to apply them to diverse fields such as finance [17], medicine [22] or engineering [24].

But, surprisingly, RL algorithms have trouble to leave the laboratories and become used in real-life applications. One possible reason for this may be the black-box nature of policies computed by current state-of-the-art RL algorithms. Indeed, when the state space is huge or continuous, policies are usually based on smart approximation structures, such as neural networks, ensembles of regression trees or linear combinations of basis functions [6]. While the use of such approximation structures often leads to algorithms providing high-precision solutions, it comes with the price of jeopardizing the interpretability by human experts of their results.

In real-world applications, interpretable policies are preferable to black-box policies for several reasons. First, when addressing a sequential decision making problem, people may be uncertain about their system model. In such a case, even an algorithm coming with strong theoretical guarantees may produce doubtful results. This lack of trust could to some extend be eluded, or reduced, if one could at least “understand” the policy. Second, in many fields, the step of formalizing the problem into an optimal sequential decision making problem involves arbitrary choices that may be somewhat disconnected from reality. The aim is then essentially to exploit techniques among the optimal sequential decision making technology that are supposed to lead to policies having desirable properties. Such properties are generally much harder to establish with black-box policies than with interpretable ones. Third, when applied *in vivo*, decisions suggested by a policy may involve extra-engineering issues (ethical, ideological, political,...) which may impose the decision process to be understandable by humans. This is especially the case in the context of medical applications involving patients' health [22, 13, 30].

Despite a rich literature in machine learning, the notion of interpretability has not yet received a satisfactory and broadly accepted formal definition. Besides this, a significant body of work has been devoted to the definition of algorithmic complexity (e.g. Kolmogorov complexity [18], its application to density estimation in [5], and the question of defining artificial intelligence in general [16]) and its implications in terms of the consistency of machine learning algorithms, but this complexity notion is language-dependent and is therefore not systematically transposable as a measure of interpretability by human experts of a hypothesis computed by a machine learning algorithm.

Given this situation, we propose in this paper a "pragmatic" three step approach for the design of interpretable reinforcement learning algorithms. The first step consists of choosing a human-readable language to represent the policies computed by an algorithm: we propose to this end a simple grammar of formulas using a restricted number of operators and terminal symbols that are used to express action-ranking indexes. The second step consists of defining a complexity measure of these formulas: to this end we use the number of nodes of the derivation tree that produces a formula from the chosen grammar. The third step consists of measuring the (non)interpretability of a policy by the complexity of its shortest representation in the formula language and by formulating a policy search problem under bounded complexity in this sense.

The rest of this paper is organized as follows. Section 2 formalizes the problem addressed in this paper. Section 3 details a particular class of interpretable policies that are implicitly defined by maximizing state-action dependent indices in the form of small, closed-form formulas. Section 4 formalizes the search of a high-performance policy in this space as a multi-armed bandit problem where each arm corresponds to a formula-based policy. This defines a direct policy search scheme for which Section 5 provides an empirical evaluations on several RL benchmarks. We show that on all benchmarks, this approach manages to compute accurate and indeed interpretable policies, that often outperform uniform planning policies of depth 10. Section 6 proposes a brief review of the RL literature dealing with the notion of interpretability and Section 7 concludes.

## 2 Problem formalization

We consider a stochastic discrete-time system whose dynamics is described by a time-invariant equation

$$x_{t+1} \sim p_f(.|x_t, u_t) \quad t = 0, 1, \ldots$$

where for all $t$, the state $x_t$ is an element of the $d_{\mathcal{X}}-$dimensional state space $\mathcal{X}$, $u_t$ is an element of the finite (discrete) $d_{\mathcal{U}}-$dimensional action space $\mathcal{U} = \left\{u^{(1)}, \ldots, u^{(m)}\right\}$ ($m \in \mathbb{N}_0$) and $p_f(.)$ denotes a probability distribution function over the space $\mathcal{X}$. A stochastic instantaneous scalar reward

$$r_t \sim p_\rho(.|x_t, u_t)$$

is associated with the action $u_t$ taken while being in state $x_t$, where $p_\rho(\cdot)$ denotes a probability distribution over rewards. Let $\mathit{\Pi}$ be the set of stochastic stationary policies, i.e. the set of stochastic mappings from $\mathcal{X}$ into $\mathcal{U}$ . Given a policy $\pi \in \mathit{\Pi}$, we denote by $\pi(x_t) \sim p_\pi(.|x_t)$ a stochastic action suggested by the policy $\pi$ in the state $x_t$. Given a probability distribution over the set of initial states $p_0(.)$, the performance of a policy $\pi$ can be defined as:

$$J^\pi = \mathop{\mathbb{E}}_{p_0(.), p_f(.), p_\rho(.)} [\mathcal{R}^\pi(x_0)]$$

where $\mathcal{R}^\pi(x_0)$ is the stochastic return of the policy $\pi$ when starting from $x_0$. The return that is often used is the infinite discounted sum of rewards:

$$\mathcal{R}^\pi(x_0) = \sum_{t=0}^{\infty} \gamma^t r_t \; ,$$

where $r_t \sim p_\rho(.|x_t, \pi(x_t))$, $x_{t+1} \sim p_f(.|x_t, \pi(x_t))$, $\pi(x_t) \sim p_\pi(.|x_t)$ $\forall t \in \mathbb{N}$ and $\gamma < 1$. Note that one can consider other criteria to evaluate the return of a trajectories, such as finite-time horizon sum of rewards, or more sophisticated criteria such as value-at-risk. An optimal policy $\pi^*$ is a policy such that

$$\forall \pi \in \mathit{\Pi}, \; J^\pi \leq J^{\pi^*}.$$

In most non-trivial RL problems, such as those involving a continuous state space $\mathcal{X}$, the policy space $\mathit{\Pi}$ cannot be represented explicitly in a machine. What RL algorithms do to overcome this difficulty is to consider a subset of policies from $\mathit{\Pi}$ that can be compactly represented, such as parametric policies or value function-based policies. In this paper, we additionally expect the policies from such a subset to be interpretable by humans.

We use the ideas of Kolmogorov complexity theory to express the interpretability of a policy $\pi$ relative to a given description language. We say that a policy is interpretable, in the selected description language, if it can be described in this language by using few symbols. This notion is rather general and can be applied to several description languages, such as decision lists, decision trees, decision graphs or more general mathematical formulas.

Given a policy $\pi$, we denote $D_L(\pi)$ the set of descriptors of $\pi$ in the chosen description language $L$. Formally, the Kolmogorov complexity of $\pi$ is the number of symbols taken by the shortest description in $D_L(\pi)$:

$$\kappa_L(\pi) = \min_{d \in D_L(\pi)} |d|.$$

The remainder of this paper proposes a policy description language in the form of mathematical formulas and addresses the problem of finding the best policy whose Kolmogorov complexity is no more than $K \in \mathbb{N}_0$ in this language:

$$\pi^*_{int} = \mathop{\arg\max}_{\{\pi \in \mathit{\Pi} | \kappa_L(\pi) \leq K\}} J^\pi.$$

## 3 Building a space of interpretable policies

We now introduce index-based policies and define the subset of low Kolmogorov complexity index-based policies that we focus on in this paper.

### 3.1 Index-based policies

Index-based policies are policies that are implicitly defined by maximizing a state-action index function. Formally, we call any mapping $I : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ a state-action index function. Given a state-action index function $I$ and a state $x \in \mathcal{X}$, a decision $\pi_I(x)$ can be taken by drawing an action in the set of actions that lead to the maximization of the value $I(x, u)$:

$$\forall x \in \mathcal{X}, \pi_I(x) \in \underset{u \in \mathcal{U}}{\arg\max} \quad I(x, u) \ .$$

Such a procedure defines a class of stochastic policies[1]. It has already been vastly used in the particular case where state-action value functions are used as index functions[2].

### 3.2 Formula-based index functions

We move on the problem of determining a subclass of low Kolmogorov complexity index functions. To this purpose, we consider index functions that are given in the form of small, closed-form formulas. Closed-form formulas have several advantages: they can be easily computed, they can formally be analyzed (e.g. differentiation, integration) and, when they are small enough, they are easily interpretable.

Let us first explicit the set of formulas $\mathbb{F}$ that we consider in this paper. A formula $F \in \mathbb{F}$ is:

– either a binary expression $F = B(F', F'')$, where $B$ belongs to a set of binary operators $\mathbb{B}$ and $F'$ and $F''$ are also formulas from $\mathbb{F}$,
– or a unary expression $F = U(F')$ where $U$ belongs to a set of unary operators $\mathbb{U}$ and $F' \in \mathbb{F}$,
– or an atomic variable $F = V$, where $V$ belongs to a set of variables $\mathbb{V}$,
– or a constant $F = C$, where $C$ belongs to a set of constants $\mathbb{C}$.

In the following, we consider a set of operators and constants that provides a good compromise between high expressiveness and low cardinality of $\mathbb{F}$. The set of binary operators considered in this paper $\mathbb{B}$ includes the four elementary mathematic operations and the $\min$ and $\max$ operators: $\mathbb{B} = \{+, -, \times, \div, \min, \max\}$ . The set of unary operators $\mathbb{U}$ contains the square root, the logarithm, the absolute value, the opposite and the inverse: $\mathbb{U} = \left\{\sqrt{.}, \ln(.), |.|, -., \frac{1}{.}\right\}$ . The set of variables $\mathbb{V}$ gathers all the available variables of the RL problem. In this paper, we consider two different settings: in the *lookahead-free setting*, we consider that index functions only depend on the current

[1] Ties are broken randomly in our experiments.
[2] State-action value functions map the pair $(x, u)$ into an estimate of the expected return when taking action $u$ in state $x$ and following a given policy afterwards.

state and action $(x_t, u_t)$. In this setting, the set of variables $\mathbb{V}$ contains all the components of $x_t$ and $u_t$:

$$\mathbb{V} = \mathbb{V}_{LF} = \left\{ x_t^{(1)}, \ldots, x_t^{(d_{\mathcal{X}})}, u_t^{(1)}, \ldots, u_t^{(d_{\mathcal{U}})} \right\} .$$

In the *one-step lookahead setting*, we assume that the probability distributions $p_f(.)$ and $p_\rho(.)$ are accessible to simulations, i.e., one can draw a value of $x_{t+1} \sim p_f(.|x_t, u_t)$ and $r_t \sim p_\rho(.|x_t, u_t)$ for any state-action pair $(x_t, u_t) \in \mathcal{X} \times \mathcal{U}$. To take advantage of this, we will consider state-action index functions that depend on $(x_t, u_t)$ but also on the outputs of the simulator $(r_t, x_{t+1})$. Hence, the set of variables $\mathbb{V}$ contains all the components of $x_t$, $u_t$, $r_t$ and $x_{t+1}$:

$$\mathbb{V} = \mathbb{V}_{OL} = \left\{ x_t^{(1)}, \ldots, x_t^{(d_{\mathcal{X}})}, u_t^{(1)}, \ldots, u_t^{(d_{\mathcal{U}})}, r_t, x_{t+1}^{(1)}, \ldots, x_{t+1}^{(d_{\mathcal{X}})} \right\} .$$

The set of constants $\mathbb{C}$ has been chosen to maximize the number of different numbers representable by small formulas. It is defined as $\mathbb{C} = \{1, 2, 3, 5, 7\}$. In the following, we abusively identify a formula with its associated index function, and we denote by $\pi_F$ the policy associated with the index function defined by the formula $F$. In other words, the policy $\pi_F$ is the myopic greedy policy w.r.t. $F$, where $F$ act as a surrogate for the long-term return.

### 3.3 Interpretable index-based policies using small formulas

Several formulas can lead to the same policy. As an example, any formula $F$ that represents an increasing mapping that only depends on $r_t$ defines the greedy policy.

Formally, given a policy $\pi$, we denote $D_F(\pi) = \{F \in \mathbb{F} | \pi_F = \pi\}$ the set of descriptor formulas of this policy. We denote $|F|$ the description length of the formula $F$, i.e. the total number of operators, constants and variables occurring in $F$. Given these notations, the Kolmogorov complexity of $\pi$ such that $D_F(\pi) \neq \emptyset$ is

$$\kappa(\pi) = \min_{F \in D_F(\pi)} |F|.$$

Let $K \in \mathbb{N}$ be a fixed maximal length. We introduce our set of interpretable policies $\Pi_{int}^K$ as the set of formula-based policies whose Kolmogorov complexity is lower or equal than $K$:

$$\Pi_{int}^K = \{\pi | D_F(\pi) \neq \emptyset \text{ and } \kappa(\pi) \leq K\} .$$

## 4 Direct policy search in a space of interpretable policies

We now focus on the problem of finding a high-performance policy $\pi_{F^*} \in \Pi_{int}^K$. For computational reasons, we approximate the set $\Pi_{int}^K$ by a set $\tilde{\Pi}_{int}^K$ using a strategy detailed in Section 4.1. We then describe our direct policy search scheme for finding a high-performance policy in the set $\tilde{\Pi}_{int}^K$ in Section 4.2.

### 4.1 Approximating $\Pi^K_{int}$

Except in the specific case where the state space is finite, computing the set $\Pi^K_{int}$ is not trivial. We propose instead to approximately discriminate between policies by comparing them on a finite sample of state points. More formally, the procedure is as following:

- we first build $\mathbb{F}^K$, the space of all formulas such that $|F| \leq K$,
- given a finite sample of $\mathcal{S}$ state points $\mathcal{S} = \{s_i\}_{i=1}^{S}$, we clusterize all fomulas from $\mathbb{F}^K$ according to the following rule: two formulas $F$ and $F'$ belong to the same cluster if

$$\forall s \in \{s_1, \ldots, s_S\},$$
$$\arg\max_{u \in \mathcal{U}} F(s,u,r,y) = \arg\max_{u \in \mathcal{U}} F'(s,u,r,y)$$

  for some realizations $r \sim p_\rho(.|s,u)$ and $y \sim p_f(.|s,u)$ (in the lookahead-free setting, the previous rule does not take $r$ and $y$ into account). Formulas leading to invalid index functions (caused for instance by division by zero or logarithm of negative values) are discarded;
- among each cluster, we select one formula of minimal length;
- we gather all the selected minimal length formulas into an approximated reduced set formulas $\tilde{\mathbb{F}}^K$ and obtain the associated approximated reduced set of policies:

$$\tilde{\Pi}^K_{int} = \{\pi_F | F \in \tilde{\mathbb{F}}^K\}.$$

In the following, we denote by $N$ the cardinality of the approximate set of policies $\tilde{\Pi}^K_{int} = \{\pi_{F_1}, \ldots, \pi_{F_N}\}$.

### 4.2 Finding a high-performance policy in $\tilde{\Pi}^K_{int}$

An immediate approach for determining a high-performance policy $\pi_{F^*} \in \tilde{\Pi}^K_{int}$ would be to draw Monte Carlo simulations in order to identify the best policies. Such an approach could reveal itself to be time-inefficient in case of spaces $\tilde{\Pi}^K_{int}$ of large cardinality.

We propose instead to formalize the problem of finding a high-performance policy in $\tilde{\Pi}^K_{int}$ as a $N-$armed bandit problem. To each policy $\pi_{F_n} \in \tilde{\Pi}^K_{int}$ ($n \in \{1, \ldots, N\}$), we associate an arm. Pulling the arm $n$ means making one trajectory with the policy $\pi_{F_n}$ on the system, i.e., drawing an initial state $x_0 \sim p_0(.)$ and applying the decisions suggested by the policy $\pi_{F_n}$ until stopping conditions are reached.

Multi-armed bandit problems have been vastly studied, and several algorithms have been proposed, such as for instance all UCB-type algorithms [3, 2]. New empirically efficient approaches have also recently been proposed in [19].

## 5 Experimental results

We empirically validate our approach on several standard RL benchmarks: the "Linear Point" benchmark (LP) initially proposed in [15], the "Left or Right" problem (LoR)

**Table 1.** Benchmark characteristics: state space and action space dimensions, number of actions, stochasticity of the system, number of variables in the lookahead-free and one-step lookahead settings, discount factor and horizon truncation.

| BENCHMARK | LP | LOR | CAR | ACR | B | HIV |
|---|---|---|---|---|---|---|
| $d_{\mathcal{X}}$ | 2 | 1 | 2 | 4 | 5 | 6 |
| $d_{\mathcal{U}}$ | 1 | 1 | 1 | 1 | 2 | 2 |
| $m$ | 2 | 2 | 2 | 2 | 9 | 4 |
| Stoch. | no | yes | no | no | yes | no |
| $\#\mathbb{V}_{LF}$ | 3 | 2 | 3 | 5 | 7 | 8 |
| $\#\mathbb{V}_{OL}$ | 6 | 4 | 6 | 10 | 13 | 15 |
| $\gamma$ | .9 | .75 | .95 | .95 | .98 | .98 |
| $T$ | 50 | 20 | 1000 | 100 | 5e4 | 300 |
| Rand. | 3.881 | 36.03 | -0.387 | 0.127e-3 | -0.17 | 2.193e6 |
| LA(1) | 3.870 | 60.34 | -0.511 | 0 | -0.359 | 1.911e6 |
| LA(5) | 5.341 | 60.39 | -0.338 | 0 | -0.358 | 2.442e9 |
| LA(10) | 5.622 | 60.45 | -0.116 | 0.127e-3 | - | 3.023e9 |
| FQI$^*$ | - | 64.3 | 0.29 | 44.7e-3 | 0 | 4.16e9 |

[8], the "Car on the Hill" problem (CAR) [21], the "Acrobot Swing Up" problem (ACR) [29], the "Bicycle balancing" problem (B) [23] and the HIV benchmark (HIV) [1]. The choice of these benchmarks was made a priori and independently of the results obtained with our methods, and no benchmark was later excluded.

We evaluate all policies using the same testing protocol as in [8]: the performance criterion is the discounted cumulative regret averaged over a set of problem-dependent initial states $\mathcal{P}_0$ (see Appendix A), estimated through Monte Carlo simulation, with $10^4$ runs per initial state and with a truncated finite horizon $T$.

Table 1 summarizes the characteristics of each benchmark, along with baseline scores obtained by the random policy and by uniform look-ahead (LA) planning policies. The LA(1) policy (resp. LA(5) and LA(10)) uses the simulator of the system to construct a look-ahead tree uniformly up to depth 1 (resp. 5 and 10). Once this tree is constructed, the policy returns the initial action of a trajectory with maximal return. Note that LA(1) is equivalent to the greedy policy w.r.t. instantaneous rewards.

When available, we also display the best scores reported in [8] for Fitted Q Iteration (FQI)[3].

## 5.1 Protocol

In the present set of experiments, we consider two different values for the maximal length of formulas: $K = 5$ and $K = 6$. For each value of $K$ and each benchmark, we

[3] Note that, while we use the same evaluating protocol, the scores relative to FQI should be taken with a grain of salt: FQI relies on the "batch-mode" RL setting, in which the trainer only has access to a finite sample of system transitions, whereas, our direct policy search algorithm can simulate the system infinitely many times. By using more simulations, the scores of FQI could probably by slightly higher than those reported here.

first build the set $\mathbb{F}^K$. We then consider a set of test points $\mathcal{S}$ that we use to extract $\tilde{\Pi}^K_{int}$ according to the procedure described in Section 4.1. When the state space is bounded and the borders of the state space are known, the set $\mathcal{S}$ is obtained by uniformly sampling 100 points within the domain. Otherwise, for unbounded problems, we refer to the literature for determining a bounded domain that contains empirical observations of previous studies. The probability distribution of initial states $p_0(.)$ used for training is also chosen uniform. Appendix A details the domains used for building $\tilde{\Pi}^K_{int}$ and those used for $p_0(.)$.

For solving the multi-armed bandit problem described in Section 4.2, we use a recently proposed bandit policy that has shown itself to have excellent empirical properties [19]. The solution works as follows: each arm is first drawn once to perform initialization. The $N$ arms are then associated with a time-dependent index $A_{n,t}$. At each time step $t \in \{0, \ldots, T_b\}$, we select and draw one trajectory with the policy $\pi_{F_n}$ whose index:

$$A_{n,t} = \bar{r}_{n,t} + \frac{\alpha}{\theta_{n,t}}$$

is maximized ($\bar{r}_{n,t}$ denotes the empirical average of all the returns that have been received when playing policy $\pi_{F_n}$, and $\theta_{n,t}$ denotes the number of times the policy $\pi_{F_n}$ has been played so far). The constant $\alpha > 0$ allows to tune the exploration/exploitation trade-off and the parameter $T_b$ represents the total budget allocated to the search of a high-performance policy. We performed nearly no tuning and used the same values of these parameters for all benchmarks: $\alpha = 2$, $T_b = 10^6$ when $K = 5$ and $T_b = 10^7$ when $K = 6$. At the end of the $T_b$ plays, policies can be ranked according to the empirical mean of their return. To illustrate our approach, we only report the best performing policies w.r.t. this criterion in the following. Note that to go further into interpretability, one could not only analyze the best performing policy but also the whole top-list of policies for better extracting common characteristics of good policies.

### 5.2 A typical run of the algorithm

In order to illustrate the behavior of the algorithm, we compute and plot in Figure 1, every 1000 iterations, the performance of the policy having the best average empirical return in the specific case of the LP benchmark in both lookahead-free and one-step lookahead settings with $K = 5$.

In the lookahead-free setting, we have $N = 907$ candidate policies, which means that all policies have been played at least once after 1000 iterations. This explains somehow why the red curves starts almost at its best level. The one-step lookahead setting involves a much larger set of candidate policies: $N = 12214$. In this case, the best policy starts to be preferred after $10^5$ iterations, which means that, in average, each policy has been experienced $\simeq 8$ times.

We performed all experiments with a 1.9 Ghz processor. The construction of the space $\tilde{\Pi}^K_{int}$ is quite fast and takes 4s and 11s for the lookahead-free and one-step lookahead settings, respectively. The computation of $\pi_{F^*}$ (and the evaluation every 1000 iterations) requires about one hour for the LP benchmark in the case $K = 5$ and 14 hours when $K = 6$ (both in the one-step lookahead setting). Our most challenging con-

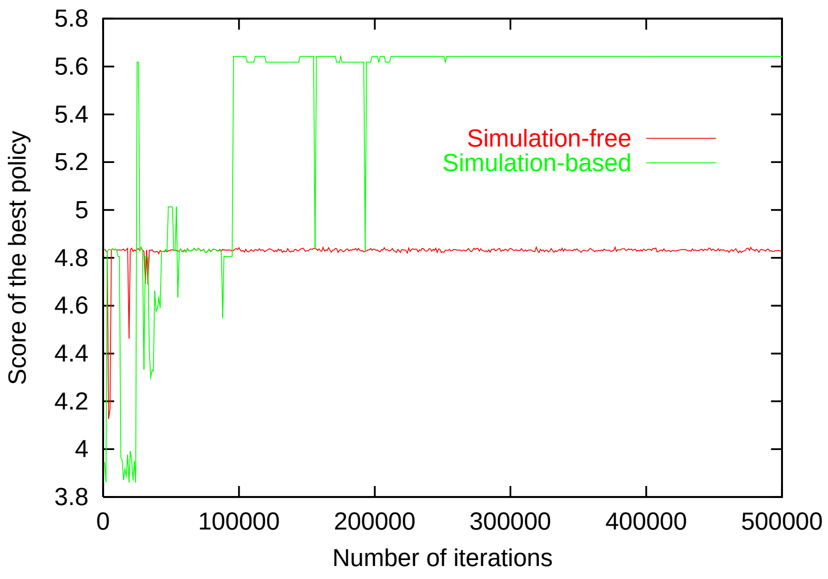

**Fig. 1.** Score of the best policy with respect to the iterations of the search algorithm on the LP benchmark.

**Table 2.** Results with $K = 5$.

| | lookahead-free | | | | one-step lookahead | | | |
|---|---|---|---|---|---|---|---|---|
| | # $\mathbb{F}^5$ | $N$ | $\hat{J}^{\pi_{F^*}}$ | $F^*$ | # $\mathbb{F}^5$ | $N$ | $\hat{J}^{\pi_{F^*}}$ | $F^*$ |
| LP | 106 856 | 907 | 4.827 | $\lvert v-a \rvert$ | 224 939 | 12 214 | 5.642 | $\lvert 1/(y+v') \rvert$ |
| LoR | 78 967 | 513 | 64.04 | $(x-2)u$ | 140 193 | 3 807 | 64.27 | $1/\sqrt{x-u}$ |
| CAR | 106 856 | 1 106 | 0.101 | $u/(2-s)$ | 224 939 | 13 251 | 0.248 | $\sqrt{r}+s'$ |
| ACR | 179 410 | 3 300 | 0.127e-3 | 1 (*random*) | 478 815 | 43 946 | 0.127e-3 | 1 (*random*) |
| B | 277 212 | 11 534 | -1.07e-3 | $\dot{\omega}/(\dot{\theta}+T)$ | 756 666 | 94 621 | 0 | $(\dot{\omega}'-d)/\dot{\theta}'$ |
| HIV | 336 661 | 5 033 | 5.232e6 | $(T_1^*-T_2)\epsilon_1$ | 990 020 | 82 944 | 3.744e9 | $\sqrt{E'}/\ln(T_1')$ |

figuration is the B benchmark in the one-step lookahead setting with $K = 6$, for which learning requires about 17 days.

## 5.3 Results

We provide in this section the results that have been obtained by our approach on the six benchmarks. In Table 2, we give the performance of the obtained formulas in both the lookahead-free and one-step lookahead settings using $K = 5$. Table 3 reports the results when using $K = 6$. For each setting, we provide the cardinality $\#\mathbb{F}^K$ of the original set of index functions based on small formulas, the cardinality $N$ of the reduced search space $\tilde{\Pi}^K_{int}$, the score $\hat{J}^{\pi_{F^*}}$ of the high-performance policy $\pi_{F^*}$ and the expression of the formula $F^*$, using the original variable names detailed in Appendix A (primes $'$ indicate next state variables).

**Table 3.** Results with $K = 6$.

| | lookahead-free | | | | one-step lookahead | | | |
|---|---|---|---|---|---|---|---|---|
| | # $\mathbb{F}^6$ | $N$ | $\hat{J}^{\pi_{F^*}}$ | $F^*$ | # $\mathbb{F}^6$ | $N$ | $\hat{J}^{\pi_{F^*}}$ | $F^*$ |
| LP | 1 533 456 | 8 419 | 5.642 | $(-y-v)a$ | 3 562 614 | 130 032 | 5.642 | $y' - \lvert y+v' \rvert$ |
| LoR | 1 085 742 | 3 636 | 64.28 | $u/(x-\sqrt{5})$ | 2 088 018 | 31 198 | 64.32 | $u/(x-\sqrt{7})$ |
| Car | 1 533 456 | 10 626 | 0.174 | $u(\sqrt{7}-s)$ | 3 562 614 | 136 026 | 0.282 | $r - \frac{1}{\max(p',s')}$ |
| Acr | 2 760 660 | 36 240 | 0.238e-3 | $\max(\dot{\theta}_2/u, \sqrt{2})$ | 8 288 190 | 548 238 | 15.7e-3 | $\dot{\theta}_2 \lvert \dot{\theta}'_2 \rvert - u$ |
| B | 4 505 112 | 132 120 | -0.36e-3 | $\psi\dot{\theta} - \lvert d \rvert$ | 13 740 516 | 1 204 809 | 0 | $1/(7-\dot{\theta}'/\dot{\omega}')$ |
| HIV | 5 559 386 | 40 172 | 5.217e6 | $1/(\epsilon_1 - \frac{T_2}{T_1^*})$ | 18 452 520 | 798 004 | 3.744e9 | $\sqrt{E'}/\ln(T'_1)$ |

**Cardinaly of $\tilde{\Pi}^K_{int}$.** The cardinality $N$ of $\tilde{\Pi}^K_{int}$ is lower than the cardinality of $\mathbb{F}^K$ up to three or four orders of magnitude. This is due to (i) the elimination of non-valid formulas, (ii) equivalent formulas and (iii) approximation errors that occur when $\mathcal{S}$ does not enable to distinguish between two nearly identical policies.

**Formula length and impact of lookahead.** For a fixed length $K$, results obtained in the one-step lookahead setting are better than those obtained in the lookahead-free setting, which was expected since $\mathbb{V}_{LF} \subset \mathbb{V}_{OL}$. Similarly, for a fixed setting (lookahead-free or one-step lookahead), we observe that results obtained in the case $K = 6$ are better than those obtained in the case $K = 5$. This result was also expected since, for a fixed setting, $\mathbb{F}^5 \subset \mathbb{F}^6$.

**Comparison with baseline policies.** For all the benchmarks, both settings with $K = 6$ manage to find interpretable policies outperforming the LA(10) baseline. For the B benchmark, we discover optimal policies (0 is the best possible return for this problem) for both one-step lookahead settings. The fact that very small index formulas enable to outperform large look-ahead trees containing $m^{10}$ nodes is quite impressive and reveals an aspect that may have been under-estimated in past RL research: many complex control problems accept simple and interpretable high-performance policies.

All our interpretable policies outperform the random policy and greedy policy (LA(1)), though, in some cases, $K = 5$ is not sufficient to outperform LA(10). As an example, consider the HIV benchmark in the lookahead-free setting: it seems impossible in this case to incorporate information on both the state and the two action dimensions using only 5 symbols. Since only one of the two action variables appears in the best formula ($\epsilon_1$), the corresponding policy is not deterministic and chooses the second action variable ($\epsilon_2$) randomly, which disables reaching high performance on this benchmark.

**Comparison with FQI.** Except for the B benchmark, for which we discovered interpretable optimal policies, $\pi_{F^*}$ policies are generally outperformed by FQI policies. This illustrates the antagonism between performance and interpretability, a well-known phenomenon in machine learning. Although our policies are outperformed by FQI, their interpretability is much higher, which may be a decisive advantage in real-world applications.

**Interpretability of obtained policies.** We first provide an illustration on how the analytical nature of formulas can be exploited to interpret the behavior of the corresponding policies. We consider the best formula obtained for the LP problem in Table 3:

$$F^* = (-y - v)a = -a(y + v).$$

Since $a$ is either equal to $-1$ or 1, we can straightforwardly compute a closed-form of the policy $\pi_{F^*}$:

$$\pi_{F^*}(y, v) = -sign(y + v).$$

In other terms, the policy selects $a = -1$ when $y > -v$ and $a = 1$ otherwise, which is extremely interpretable.

We now focus on the formula obtained for the HIV benchmark:

$$F^* = \frac{\sqrt{E'}}{\ln(T_1')}.$$

This policy depends on both the concentration $E$ of cytotoxic T-lymphocytes (in cells/ml) and on the concentration $T_1$ of non-infected CD4 T-lymphocytes (in cells/ml) (both taken at the subsequent stage). The first category of cells corresponds to the specific immunological response to the HIV infection whereas the second category of cells is the main target of HIV. Maximizing the formula amounts in boosting the specific immunological response against HIV without increasing too much the concentration $T_1$ which favors the HIV replication. We believe that such kind of results may be of major interest for the medical community.

## 6 Related work

While interpretability is a concern that has raised a lot of interest among the machine learning community (e.g. [28, 26]), it has surprisingly not been addressed so much in the RL community. However, works dealing with feature discovery [11], variable selection [14, 9, 7] or dimensionality reduction in RL [4] can indeed be considered as first steps towards interpretable solutions.

The work proposed in this paper is also related to approaches aiming to derive optimization schemes for screening policy spaces, such as gradient-free techniques using cross-entropy optimization [25, 20], genetic algorithms [12] and more specifically related to our work, genetic programming algorithms [27, 10].

Finally, our approach is closely related to the work of [19] which proposes to automatically discover efficient indices - given in the form of small formulas - for solving multi-armed bandit problems.

## 7 Conclusions

In this paper, we have proposed an approach for inferring interpretable policies to RL problems. We have focused on the case where interpretable solutions are provided by index-based policies computed from small, closed-form formulas. The problem of identifying a high-performance formula-based policy was then formalized as a multi-armed bandit problem. Although promising empirical results have been obtained on standard

RL benchmarks, we have also experienced the antagonism between optimality and interpretability, a well known problem in machine learning.

In this paper, we have focused on a very specific class of interpretable solutions using small formulas expressed in a specific grammar. But one could also imagine searching in other types of interpretable policy spaces based on simple decision trees or graphs. Another direct extension of this work would be to consider RL problems with continuous actions. In this case, we could try to directly search for formulas computing the values of the recommended actions.

## Acknowledgements

This paper presents research results of the Belgian Network BIOMAGNET funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. Raphael Fonteneau is a Post-doctoral fellow of the FRS-FNRS (Funds for Scientific Research). The authors also thank the PASCAL2 European Network of Excellence.

# Bibliography

[1] Adams, B., Banks, H., Kwon, H.D., Tran, H.: Dynamic multidrug therapies for HIV: Optimal and STI approaches. Mathematical Biosciences and Engineering 1, 223–241 (2004)

[2] Audibert, J., Munos, R., Szepesvári, C.: Tuning bandit algorithms in stochastic environments. In: Algorithmic Learning Theory. pp. 150–165. Springer (2007)

[3] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine learning 47(2), 235–256 (2002)

[4] Bar-Gad, I., Morris, G., Bergman, H.: Information processing, dimensionality reduction and reinforcement learning in the basal ganglia. Progress in Neurobiology 71(6), 439–473 (2003)

[5] Barron, A.R., Cover, T.M.: Minimum complexity density estimation. Information Theory, IEEE Transactions on 37(4), 1034–1054 (1991)

[6] Busoniu, L., Babuska, R., De Schutter, B., Ernst, D.: Reinforcement Learning and Dynamic Programming using Function Approximators. Taylor & Francis CRC Press (2010)

[7] Castelletti, A., Galelli, S., Restelli, M., Soncini-Sessa, R.: Tree-based variable selection for dimensionality reduction of large-scale control systems. In: Adaptive Dynamic Programming and Reinforcement Learning (ADPRL). pp. 62–69. IEEE (2011)

[8] Ernst, D., Geurts, P., Wehenkel, L.: Tree-based batch mode reinforcement learning. Journal of Machine Learning Research 6, 503–556 (2005)

[9] Fonteneau, R., Wehenkel, L., Ernst, D.: Variable selection for dynamic treatment regimes: a reinforcement learning approach. In: European Workshop on Reinforcement Learning (EWRL) (2008)

[10] Gearhart, C.: Genetic programming as policy search in markov decision processes. Genetic Algorithms and Genetic Programming at Stanford pp. 61–67 (2003)

[11] Girgin, S., Preux, P.: Feature discovery in reinforcement learning using genetic programming. In: 11th European Conference on Genetic Programming. pp. 218–229. Springer-Verlag (2008)

[12] Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-wesley (1989)

[13] Guez, A., Vincent, R., Avoli, M., Pineau, J.: Adaptive treatment of epilepsy via batch-mode reinforcement learning. In: Innovative Applications of Artificial Intelligence (IAAI). pp. 1671–1678 (2008)

[14] Gunter, L., Zhu, J., Murphy, S.: Artificial Intelligence in Medicine., vol. 4594/2007, chap. Variable Selection for Optimal Decision Making, pp. 149–154. Springer Berlin / Heidelberg (2007)

[15] Hren, J., Munos, R.: Optimistic planning of deterministic systems. Recent Advances in Reinforcement Learning pp. 151–164 (2008)

[16] Hutter, M.: Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability. Springer, Berlin (2005)

[17] Ingersoll, J.: Theory of Financial Decision Making. Rowman and Littlefield Publishers, Inc. (1987)
[18] Kolmogorov, A.N.: Three approaches to the quantitative definition of information. Problems of Information Transmission 1(1), 1–7 (1965)
[19] Maes, F., Wehenkel, L., Ernst, D.: Automatic discovery of ranking formulas for playing with multi-armed bandits. In: 9th European workshop on reinforcement learning (EWRL). Athens, Greece (September 2011)
[20] Maes, F., Wehenkel, L., Ernst, D.: Optimized look-ahead tree search policies. In: 9th European workshop on reinforcement learning (EWRL). Athens, Greece (September 2011)
[21] Moore, A., Atkeson, C.: The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. Machine Learning 21(3), 199–233 (1995)
[22] Murphy, S.: Optimal dynamic treatment regimes. Journal of the Royal Statistical Society, Series B 65(2), 331–366 (2003)
[23] Randløv, J., Alstrøm, P.: Learning to drive a bicycle using reinforcement learning and shaping. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML). pp. 463–471. Citeseer (1998)
[24] Riedmiller, M.: Neural fitted Q iteration - first experiences with a data efficient neural reinforcement learning method. In: Proceedings of the Sixteenth European Conference on Machine Learning (ECML). pp. 317–328. Porto, Portugal (2005)
[25] Rubinstein, R., Kroese, D.: The Cross-Entropy Method. A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning. Information Science and Statistics, Springer (2004)
[26] Rüping, S.: Learning Interpretable Models. Ph.D. thesis (2006)
[27] Stanley, K., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10(2), 99–127 (2002)
[28] Wehenkel, L.: Automatic Learning Techniques in Power Systems. Kluwer Academic, Boston (1998)
[29] Yoshimoto, J., Ishii, S., Sato, M.: Application of reinforcement learning to balancing of acrobot. In: Systems, Man, and Cybernetics Conference Proceedings. vol. 5, pp. 516–521. IEEE (1999)
[30] Zhao, Y., Kosorok, M., Zeng, D.: Reinforcement learning design for cancer clinical trials. Statistics in Medicine 28, 3294–3315 (2009)

## A Parameters $p_0(.)$, $\mathcal{S}$ and $\mathcal{P}_0$

Table 4 details for each benchmark, the original name of the state and action variables, the domain $\mathcal{S}$ used for discriminating between formulas when building the set $\tilde{\Pi}^K_{int}$, the domain defining the uniform training distribution $p_0(.)$ and the set of testing initial states $\mathcal{P}_0$. The first problem LP is formally defined in [15] and $\mathcal{P}_0$ is uniform grid over the domain. We use the LoR, Acr, Car and B benchmarks as defined in the appendices of [8], with the same testing initial states $\mathcal{P}_0$ as them. The HIV benchmark is formally defined in [1] and we use a single testing initial state, known as the "unhealthy locally stable equilibrium point".

**Table 4.** Domains

| State var. | Name | $\mathcal{S}$ | $p_0(.)$ | $\mathcal{P}_0$ | Action var. | Name | $\mathcal{U}$ |
|---|---|---|---|---|---|---|---|
| Linear Point (LP) | | | | | | | |
| $x^{(1)}$ | $y$ | $[-1, 1]$ | $[-1, 1]$ | $\{-1, -0.8, \dots, 1\}$ | $u^{(1)}$ | $a$ | $\{-1, 1\}$ |
| $x^{(2)}$ | $v$ | $[-2, 2]$ | $[-2, 2]$ | $\{-2, -1.6, \dots, 2\}$ | | | |
| Left or Right (LoR) | | | | | | | |
| $x^{(1)}$ | $x$ | $[0, 10]$ | $[0, 10]$ | $\{0, 1, \dots, 10\}$ | $u^{(1)}$ | $u$ | $\{-2, 2\}$ |
| Car on the Hill (Car) | | | | | | | |
| $x^{(1)}$ | $p$ | $[-1, 1]$ | $[-1, 1]$ | $\{-1, -0.875, \dots, 1\}$ | $u^{(1)}$ | $u$ | $\{-4, 4\}$ |
| $x^{(2)}$ | $s$ | $[-3, 3]$ | $[-3, 3]$ | $\{-3, -2.625, \dots, 3\}$ | | | |
| Acrobot Swing Up (Acr) | | | | | | | |
| $x^{(1)}$ | $\theta_1$ | $[-\pi, \pi]$ | $[-2, 2]$ | $\{-2, -1.9, \dots, 2\}$ | $u^{(1)}$ | $u$ | $\{-5, 5\}$ |
| $x^{(2)}$ | $\dot{\theta}_1$ | $[-10, 10]$ | $\{0\}$ | $\{0\}$ | | | |
| $x^{(3)}$ | $\theta_2$ | $[-\pi, \pi]$ | $\{0\}$ | $\{0\}$ | | | |
| $x^{(4)}$ | $\dot{\theta}_2$ | $[-10, 10]$ | $\{0\}$ | $\{0\}$ | | | |
| Bicycle balancing (B) | | | | | | | |
| $x^{(1)}$ | $\omega$ | $[-\frac{\pi}{15}, \frac{\pi}{15}]$ | $\{0\}$ | $\{0\}$ | $u^{(1)}$ | $d$ | $\{-0.02, 0, 0.02\}$ |
| $x^{(2)}$ | $\dot{\omega}$ | $[-10, 10]$ | $\{0\}$ | $\{0\}$ | $u^{(2)}$ | $T$ | $\{-2, 0, 2\}$ |
| $x^{(3)}$ | $\theta$ | $[-\frac{4\pi}{7}, \frac{4\pi}{7}]$ | $\{0\}$ | $\{0\}$ | | | |
| $x^{(4)}$ | $\dot{\theta}$ | $[-10, 10]$ | $\{0\}$ | $\{0\}$ | | | |
| $x^{(5)}$ | $\psi$ | $[-\pi, \pi]$ | $[-\pi, \pi]$ | $\{-\pi, -\frac{3\pi}{4}, \dots, \pi\}$ | | | |
| HIV | | | | | | | |
| $x^{(1)}$ | $T_1$ | $[1, 10^6]$ | $[13000, 20000]$ | $\{163573\}$ | $u^{(1)}$ | $\epsilon_1$ | $\{0, 0.7\}$ |
| $x^{(2)}$ | $T_2$ | $[1, 10^6]$ | $[4, 6]$ | $\{5\}$ | $u^{(2)}$ | $\epsilon_2$ | $\{0, 0.3\}$ |
| $x^{(3)}$ | $T_1^*$ | $[1, 10^6]$ | $[9500, 14500]$ | $\{11945\}$ | | | |
| $x^{(4)}$ | $T_2^*$ | $[1, 10^6]$ | $[37, 55]$ | $\{46\}$ | | | |
| $x^{(5)}$ | $V$ | $[1, 10^6]$ | $[51000, 77000]$ | $\{76702\}$ | | | |
| $x^{(6)}$ | $E$ | $[1, 10^6]$ | $[19, 29]$ | $\{24\}$ | | | |