Preference-based Evolutionary Direct Policy Search*

Róbert Busa-Fekete¹, Balázs Szörényi², Paul Weng³, Weiwei Cheng¹ and Eyke Hüllermeier¹

Abstract—We present a novel approach to preference-based reinforcement learning, namely a preference-based variant of a direct policy search method based on evolutionary optimization. The core of our approach is a preference-based racing algorithm that selects the best among a given set of candidate policies with high probability. To this end, the algorithm operates on a suitable ordinal preference structure and only uses pairwise comparisons between sample rollouts of the policies. We present first experimental studies showing that our approach performs well in practice.

I. INTRODUCTION

Preference-based reinforcement learning (PBRL) is a novel research direction combining reinforcement learning (RL) and preference learning [1]. It aims at extending existing RL methods so as to make them amenable to training information and external feedback more general than numerical rewards, which are often difficult to obtain or expensive to compute. For example, what is the cost of a patient’s death in a medical treatment?

Akrour et al. [2] and Cheng et al. [3] tackle the problem of learning policies solely on the basis of pairwise comparisons between trajectories, suggesting that one system behavior is preferred to another one but without committing to precise numerical rewards. Building on novel methods for preference learning, this is accomplished by providing the RL agent with qualitative policy models, such as ranking functions. More specifically, Cheng at al. train a model that ranks actions given state, using a method called label ranking. Their approach generalizes classification-based approximate policy iteration [4]. Instead of ranking actions given states, Akrour et al. exploit preferences on trajectories in order to learn a model that ranks complete policies.

In this paper, we present a preference-based extension of evolutionary direct policy search (EDPS) as proposed by Heidrich-Meisner and Igel [5]. As a direct policy search method, it shares commonalities with the approach by Akrour et al. [2], but also differs in several respects. In particular, their approach (as well as follow-up work such as [6]) is arguably more specialized and tailored for applications in robotics, in which a user interacts with the learner in an iterative process. Moreover, policy search is not performed in a parametrized policy space directly but in a feature space capturing important background knowledge about the task to be solved.

EDPS casts policy learning as a search problem in a parametric policy space, where the function to be optimized is a performance measure like expected total reward, and evolution strategies (ES) such as CMA-ES [7] are used as optimizers. Moreover, since the evaluation of a policy can only be done approximately, namely in terms of a finite number of rollouts, the authors make use of racing algorithms to control this number in an adaptive manner. These algorithms return a sufficiently reliable ranking over the current set of policies (candidate solutions), which is then used by the ES for updating its parameters and population. A key idea of our approach is to extend EDPS by replacing the value-based racing algorithm with a preference-based one. Correspondingly, the development of a preference-based racing algorithm can be seen as a core contribution of this paper.

In the next section, we briefly overview the EDPS framework for policy learning. Our preference-based generalization of this framework is introduced in Section III. Experiments are presented in Section IV, and Section V concludes the paper.

II. EVOLUTIONARY DIRECT POLICY SEARCH (EDPS)

We briefly outline the evolutionary direct policy search (EDPS) approach in Markov Decision Processes (MDP) as introduced in [5]. Assume a parametric policy space \( \Pi = \{ \pi_\Theta | \Theta \in \mathbb{R}^p \} \) to be given, where \( \Theta \) is the parameter vector. Searching a good policy can be seen as an optimization problem where the search space is the parameter space and the target function is a policy performance evaluation, such as expected total reward. To solve this optimization task, Heidrich-Meisner and Igel [5] make use of evolution strategies [8], hence the name EDPS.

Evolution strategies in general iterate the following steps:

1. Generate a population of candidate solutions (in this case, a set of policies with different parameters).
2. Evaluate the candidate solutions (estimate the performance of the policies based on simulations/histories).
3. Select the best \( \mu \) individuals based on their fitness and use them to seed the next generation.

¹This work was supported by the German Research Foundation (DFG), as part of the Priority Programme 1527 (Autonomous Learning) and by the ANR-10-BLAN-0215 grant of the French National Research Agency.

²Computational Intelligence Group, Department of Mathematics and Computer Science, University of Marburg, Germany busazobi,cheng,eyke}@mathematik.uni-marburg.de

³INRIA Lille - Nord Europe, Sequel project, 40 avenue Halley, 59650 Villeneuve d’Ascq, France szorenyi1@inf.u-szeged.hu

¹FR - INRIA - University of Lille, Laboratoire de Computer Science of Paris 6, University Pierre and Marie Curie, 4 place Jussieu, 75005 Paris, France paul.weng@lip6.fr

— We present a novel approach to preference-based reinforcement learning, namely a preference-based variant of a direct policy search method based on evolutionary optimization. The core of our approach is a preference-based racing algorithm that selects the best among a given set of candidate policies with high probability. To this end, the algorithm operates on a suitable ordinal preference structure and only uses pairwise comparisons between sample rollouts of the policies. We present first experimental studies showing that our approach performs well in practice.

I. INTRODUCTION

Preference-based reinforcement learning (PBRL) is a novel research direction combining reinforcement learning (RL) and preference learning [1]. It aims at extending existing RL methods so as to make them amenable to training information and external feedback more general than numerical rewards, which are often difficult to obtain or expensive to compute. For example, what is the cost of a patient’s death in a medical treatment?

Akrour et al. [2] and Cheng et al. [3] tackle the problem of learning policies solely on the basis of pairwise comparisons between trajectories, suggesting that one system behavior is preferred to another one but without committing to precise numerical rewards. Building on novel methods for preference learning, this is accomplished by providing the RL agent with qualitative policy models, such as ranking functions. More specifically, Cheng at al. train a model that ranks actions given state, using a method called label ranking. Their approach generalizes classification-based approximate policy iteration [4]. Instead of ranking actions given states, Akrour et al. exploit preferences on trajectories in order to learn a model that ranks complete policies.

In this paper, we present a preference-based extension of evolutionary direct policy search (EDPS) as proposed by Heidrich-Meisner and Igel [5]. As a direct policy search method, it shares commonalities with the approach by Akrour et al. [2], but also differs in several respects. In particular, their approach (as well as follow-up work such as [6]) is arguably more specialized and tailored for applications in robotics, in which a user interacts with the learner in an iterative process. Moreover, policy search is not performed in a parametrized policy space directly but in a feature space capturing important background knowledge about the task to be solved.

EDPS casts policy learning as a search problem in a parametric policy space, where the function to be optimized is a performance measure like expected total reward, and evolution strategies (ES) such as CMA-ES [7] are used as optimizers. Moreover, since the evaluation of a policy can only be done approximately, namely in terms of a finite number of rollouts, the authors make use of racing algorithms to control this number in an adaptive manner. These algorithms return a sufficiently reliable ranking over the current set of policies (candidate solutions), which is then used by the ES for updating its parameters and population. A key idea of our approach is to extend EDPS by replacing the value-based racing algorithm with a preference-based one. Correspondingly, the development of a preference-based racing algorithm can be seen as a core contribution of this paper.

In the next section, we briefly overview the EDPS framework for policy learning. Our preference-based generalization of this framework is introduced in Section III. Experiments are presented in Section IV, and Section V concludes the paper.

II. EVOLUTIONARY DIRECT POLICY SEARCH (EDPS)

We briefly outline the evolutionary direct policy search (EDPS) approach in Markov Decision Processes (MDP) as introduced in [5]. Assume a parametric policy space \( \Pi = \{ \pi_\Theta | \Theta \in \mathbb{R}^p \} \) to be given, where \( \Theta \) is the parameter vector. Searching a good policy can be seen as an optimization problem where the search space is the parameter space and the target function is a policy performance evaluation, such as expected total reward. To solve this optimization task, Heidrich-Meisner and Igel [5] make use of evolution strategies [8], hence the name EDPS.

Evolution strategies in general iterate the following steps:

1. Generate a population of candidate solutions (in this case, a set of policies with different parameters).
2. Evaluate the candidate solutions (estimate the performance of the policies based on simulations/histories).
3. Select the best \( \mu \) individuals based on their fitness and use them to seed the next generation.

¹This work was supported by the German Research Foundation (DFG), as part of the Priority Programme 1527 (Autonomous Learning) and by the ANR-10-BLAN-0215 grant of the French National Research Agency.

²Computational Intelligence Group, Department of Mathematics and Computer Science, University of Marburg, Germany busazobi,cheng,eyke}@mathematik.uni-marburg.de

³INRIA Lille - Nord Europe, Sequel project, 40 avenue Halley, 59650 Villeneuve d’Ascq, France szorenyi1@inf.u-szeged.hu

¹FR - INRIA - University of Lille, Laboratoire de Computer Science of Paris 6, University Pierre and Marie Curie, 4 place Jussieu, 75005 Paris, France paul.weng@lip6.fr
From a practical point of view, the number of simulations in the second step is crucial: On the one hand, the learning process gets slow if it is large, while on the other hand, the ranking over the current population is not reliable enough if the number of rollouts is too small; in that case, there is a danger of selecting a suboptimal subset of the offspring population instead of the best μ ones. Therefore, Heidrich-Meisner and Igel [5] propose to apply an adaptive uncertainty handling scheme, called \textit{racing algorithm}, for controlling the size of rollout sets in an adaptive manner.

A. Racing

The racing framework is an uncertainty handling scheme introduced in [9], [10]. Given \( K \) random variables with finite expected values, the goal is to select the \( μ \) best ones, i.e., those having the highest expected value, with probability at least \( 1 - \delta \). In addition, there is an upper bound \( n_{\text{max}} \) on the number of realizations a random variable is allowed to sample. For example, the Hoeffding race algorithm constructs confidence bounds for the empirical mean estimates based on the Hoeffding bound [11] and eliminates those random variables from sampling that are either among the best \( μ \) ones or among the worst \( K - μ \) ones with high probability. The elimination rule based on the confidence intervals can be specified as follows: If the upper confidence bound for a particular random variable is smaller than the lower bound of \( K - μ \) random variables, then it can be discarded with high probability; the inclusion of a random variable can be decided analogously.

Regarding EDPS, the random samples correspond to the outcomes of the simulations (e.g., the sum of rewards incurred following a policy) and the means to be estimated are indeed the performances of the polices in terms of the performance evaluation used. From this point of view, doing a simulation in an MDP by following policy \( π \) is equivalent to drawing an example from a probability distribution \( P_π \). Consequently, a policy along with an MDP and initial distribution can simply be seen as a random variable. Therefore, we shall subsequently consider the problem of comparing random variables that are denoted by \( X_1, \ldots, X_K \).

III. PREFERENCE-BASED EDPS

The preference-based policy learning settings considered in [12], [2] proceed from a (possibly partial) preference relation \( \prec \) over histories, and the goal is to find a policy which tends to generate preferred histories with high probability. In this regard, it is notable that, in the EDPS framework, the precise values of the function to be optimized (in this case the expected total rewards) are actually not used by the evolutionary optimizer. Instead, for seeding the next generation, the ES only needs the \textit{ranking} of the candidate solutions. The values are only used by the racing algorithm in order to produce this ranking. Consequently, an obvious approach to realizing the idea of a purely preference-based version of evolutionary direct policy search (PB-EDPS) is to replace the original racing algorithm (line step 3) by a preference-based racing algorithm that only uses pairwise comparisons between policies (or, more specifically, sample histories generated from these policies). We introduce a racing algorithm of this kind in Section III-A.

A main prerequisite of such an algorithm is a “lifting” of the preference relation \( \prec \) on the space of histories to a preference relation \( \preceq \) on the space of policies; in fact, without a relation of that kind, the problem of ranking policies is not even well-defined.

A natural definition of the preference relation \( \preceq \) that we shall adopt in this paper is as follows:

\[
X \preceq Y \ \text{if and only if} \ \mathbf{P}(Y \prec X) < \mathbf{P}(X \prec Y),
\]

where \( \mathbf{P}(Y \prec X) \) denotes the probability that the realization of \( X \) is preferred (with respect to \( \prec \)) to the realization of \( Y \). Despite the appeal of \( \preceq \) as an ordinal decision model, this relation is not necessarily transitive and may even have cycles [13]. Due to preferential cycles, the (racing) problem of selecting the \( \mu \) best options may still not be well-defined for \( \preceq \) as the underlying preference relation. To overcome this difficulty, we refer to the \textit{Copeland relation} \( \preceq_C \) as a surrogate. For a set \( X = \{X_1, \ldots, X_K\} \) of random variables, it is defined as follows [14]: \( X_i \preceq_C X_j \) if and only if \( d_i < d_j \), where \( d_i = \#\{k : X_k \prec X_i, X_k \in X\} \). Its interpretation is again simple: an option \( X_i \) is preferred to \( X_j \) whenever \( X_i \) “beats” (w.r.t. \( \preceq \)) more options than \( X_j \) does. Since the preference relation \( \preceq_C \), which is “contextualized” by the set \( X \) of random variables, has a numeric representation in terms of the \( d_i \), it is a total preorder.

A. Preference-based Racing

Our preference-based racing (PBR) setup assumes \( K \) random variables \( X_1, \ldots, X_K \) with distributions \( P_{X_1}, \ldots, P_{X_K} \), respectively, and these random variables take values in a partially ordered set \( (\Omega, \prec) \). The goal of our PBR algorithm is to find the best \( \mu \) random variables with respect to the surrogate decision model \( \preceq_C \) introduced in Section III. This leads to the following optimization task:

\[
\sum_{i \in I} \sum_{j \neq i} I\{X_j \preceq X_i\} \rightarrow \max_{I \subseteq [K] : |I| = \mu}
\]

Our solution to this optimization task under uncertainty loops over the following steps:

1. Draw samples from each random variables that are active
2. Calculate \( \hat{s}_{i,j} = \frac{1}{n_i n_j} \sum_{\ell=1}^{n_i} \sum_{e=1}^{n_j} I\{x^{(\ell)}_i \prec x^{(e)}_j\} \) where \( \{x^{(1)}_i, \ldots, x^{(n_i)}_i\} \) is the sample set drawn from \( X_i \) so far
3. Calculate confidence interval \( c_{i,j} \) for \( \hat{s}_{i,j} \) by using Hoeffding bound (with a \( \delta \) confidence parameter that is given by the user)
4. Define \( d_i = \{j : \hat{s}_{i,j} - c_{i,j} > 1/2\} \) that is a lower bound of \( d_i \)
5. Based on \( d_i \), one can eliminate some \( X_1, \ldots, X_K \) form sampling based on similar rules like in the value based case (see section II-A)

It is clear that \( P(X_i \prec X_j) \approx \hat{s}_{i,j} \) based on finite sample sets.
6) If the number of iterations is bigger than \( n_{\text{max}} \) then stop and return the current approximation of Copeland’s ranking based on \( d_{1}, \ldots, d_{K} \).

One can show that the PBR algorithm returns the best \( \mu \) random variables with respect to the surrogate decision model \( \ll_C \) with high probability if \( n_{\text{max}} \) is set big enough. What is more, an expected sample complexity analysis can be done based on Even-Dar et al. [15].

### IV. Experiments on Medical Treatment Design

Here, we tackle a problem that has been used in previous work on preference-based RL [3], [6], namely the medical treatment design for cancer clinical trials. The problem is to learn an optimal treatment policy \( \pi \) mapping states \( s = (S, \bar{X}) \in \mathcal{S} = \mathbb{R}^{2}_{+} \), where \( S \) is the tumor size and \( \bar{X} \) the toxicity (inversely related to the wellness) of the patient, to actions in the form of a dosage level \( d \in [0, 1] \); the drug is given once a month, and a patient is simulated over a fixed time horizon of six and twelve months. A corresponding simulation model (based on first-order difference equations) was originally introduced in [16].

As argued by Cheng et al. [3], the numerical rewards assigned to different health states of a patient (including the extreme case of death) are quite arbitrary in this model. Therefore, the authors propose an alternative formalization, in which histories are compared in a qualitative way: \( h' \preceq h \) if the patient survives in \( h' \) but not in \( h \), and both histories are incomparable (\( h' \perp h \)) if the patient does neither survive in \( h' \) nor in \( h \). Otherwise, if the patient survives in both histories, let \( C_X \) and \( C'_X \) denote, respectively, the maximal toxicity during the 6 and 12 months of treatment in \( h \) and \( h' \), and \( C_S \) and \( C'_S \) the respective size of the tumor at the end of the therapy. Then, preference is defined via Pareto dominance: \( h' \preceq h \) if (and only if) \( C_X \leq C'_X \) and \( C_S \leq C'_S \).

Let us again emphasize that \( \preceq \) thus defined, as well as the induced strict order \( \prec \), are only partial order relations. We used the same experimental setup, except for adding Gaussian noise \( \mathcal{N}(0, 0.01) \) to the state observation [17], thereby making the underlying MDP partially observable.

We run the implementation of [5] with the Hoeffding race algorithm and CMA-ES [7]; we refer to this implementation as EDPS. We set \( \lambda = 6 \) and \( \mu = 3 \) according to [7]. The initial global step size in CMA-ES was selected from \{0.1, 1.5, 10, 15, 25, 50, 100\}. The racing algorithm has two hyperparameters, the confidence term \( \delta \) and the maximum number of samples allowed for a single option, \( n_{\text{max}} \). We optimized \( \delta \) in the range \{0.01, 0.05, 0.1\}, while \( n_{\text{max}} \) was initialized with 40 and then adapted using the technique of [5]. All parameter values were determined by means of grid search, repeating the training process in each grid point (parameter setting) 100 times, and evaluating each model on 300 patients in terms of expected utility; we found \( \sigma_0 = 2, \delta = 0.1 \) to be optimal.

Our preference-based variant PB-EDPS as introduced in Section [III] was run with the same parameters. We used a sigmoidal policy space defined as \( \pi_{\theta}(s) = 1/(1 + \exp(-\Theta^T s)) \). As baseline methods, we run the discrete uniform random policy (randomly choosing a dosage \( d \in D' = \{0.1, 0.4, 0.7, 1.0\ \text{each month}\} \) and the constant policies that take the same dosage \( d \in D' \) independently of the patient’s health state. As a more sophisticated baseline, we furthermore used SARSÅ(\( \lambda \)) [18] with discrete action set according to the original setup[17]. Finally, we included the preference-based policy iteration (PBPI) method of [12] with the parameters reported by the authors. Each policy learning method was run until reaching a limit 5000 training episodes.

We evaluated each policy on 300 virtual patients and derived averages for \( C_X \), the maximum toxicity level, as well as \( C_S \), the tumor size at the end of the treatment. We repeated this process 100 times for each policy search.

---

\( \text{---We used an} \ \epsilon\text{-greedy policy for exploration. Initially, the learning rate} \ \alpha, \ \text{the exploration term} \ \epsilon, \ \text{and the parameter of the replacing traces} \ \lambda \ \text{were set to} \ 0.1, 0.2, \ \text{and} \ 0.05 \ \text{respectively, and decreased gradually with a decay factor} \ 1/(2^t) \ \text{where} \ \tau \ \text{is the number of training episodes. We discretized each dimension of the state space into 20 bins and used a tile coding to represent the action-value function. We refer to} \ [19] \ \text{for more details.} \)
method. Then, we plotted its mean and the 95% confidence regions (assuming a multivariate normal distribution), which represent the uncertainty coming from the repetitions of the training process. As can be seen in Figure 1, our approach is performing quite well and lies on the Pareto front of all methods (which remains true when adding the death rate, reported in the same figure, as a third criterion).

V. Conclusion and Future Work

By introducing a preference-based extension of evolutionary direct policy search, called PB-EDPS, this paper contributes to the emerging field of preference-based reinforcement learning. Our method, which merely requires qualitative comparisons between sample histories as training information (and even allows for incomparability), is based on a theoretically sound decision-theoretic framework and shows promising results in first experimental studies. The idea of preference-based racing should not be limited to reinforcement learning; instead, it seems worthwhile to explore it for other applications, too, such as multi-objective optimization with several competing objectives [20].

References