Preference-based Evolutionary Direct Policy Search*

Róbert Busa-Fekete[1], Balázs Szörényi[2], Paul Weng[3], Weiwei Cheng[1] and Eyke Hüllermeier[1]

***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)[1] as introduced in [5]. Assume a parametric policy space $\Pi = \{\pi_\Theta \,|\, \Theta \in \mathbb{R}^p\}$ to be given, where Θ 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 μ individuals based on their fitness and use them to seed the next generation.

*This 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.

[1]Computational Intelligence Group, Department of Mathematics and Computer Science, University of Marburg, Germany busarobi,cheng,eyke}@mathematik.uni-marburg.de

[2]INRIA Lille - Nord Europe, SequeL project, 40 avenue Halley, 59650 Villeneuve d'Ascq, France szorenyi@inf.u-szeged.hu

[3]Laboratory of Computer Science of Paris 6, University Pierre and Marie Curie, 4 place Jussieu, 75005 Paris, France paul.weng@lip6.fr

[1]We will use the standard notation for MDPs, thus it is a 4-tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathbf{P}, r)$, where $\mathcal{S}$ is the (possibly infinite) state space and $\mathcal{A}$ the (possibly infinite) set of actions. $\mathbf{P} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ is the transition probability that defines the random transitions $\mathbf{s}' \sim \mathbf{P}(\cdot \,|\, \mathbf{s}, a)$ from a state $\mathbf{s}$ applying the action a, and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the *reward function*, i.e., $r(\mathbf{s}, a)$ defines the reward for taking action $a \in \mathcal{A}$ in state $\mathbf{s} \in \mathcal{S}$.

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 *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 $n_{\max}$ 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 $K-\mu$ 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 variables is smaller than the lower bound of $K-\mu$ 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 $\mathbf{P}_\pi$. 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 $X_1, \cdots, X_K$.

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 *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 $\ll$ 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 $\ll$ that we shall adopt in this paper is as follows:

$$X \ll Y \text{ if and only if } \mathbf{P}(Y \prec X) < \mathbf{P}(X \prec Y) \ ,$$

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 $\ll$ 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 μ best options may still not be well-defined for $\ll$ as the underlying preference relation. To overcome this difficulty, we refer to the *Copeland relation* $\ll_C$ as a surrogate. For a set $\mathcal{X} = \{X_1, \ldots, X_K\}$ of random variables, it is defined as follows [14]: $X_i \ll_C X_j$ if and only if $d_i < d_j$, where $d_i = \#\{k : X_k \ll X_i, X_k \in \mathcal{X}\}$. Its interpretation is again simple: an option X_i is preferred to X_j whenever X_i "beats" (w.r.t. $\ll$) more options than X_j does. Since the preference relation $\ll_C$, which is "contextualized" by the set $\mathcal{X}$ 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 $\mathbf{P}_{X_1}, \ldots, \mathbf{P}_{X_K}$, 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 μ random variables with respect to the surrogate decision model $\ll_C$ introduced in Section III. This leads to the following optimization task:

$$\sum_{i \in I} \sum_{j \neq i} \mathbb{I}\{X_j \ll X_i\} \longrightarrow \max_{I \subseteq [K]:\ |I| = \mu} \tag{1}$$

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 $\widehat{s}_{i,j} = \frac{1}{n_i n_j} \sum_{\ell=1}^{n_i} \sum_{\ell'=1}^{n_j} \mathbb{I}\{x_i^{(\ell)} \prec x_j^{(e')}\}$ where $\{x_i^{(1)}, \ldots, x_i^{(n_i)}\}$ is the sample set drawn from X_i so far[2].
3) Calculate confidence interval $c_{i,j}$ for $\widehat{s}_{i,j}$ by using Hoeffding bound (with a δ confidence parameter that is given by the user)
4) Define $\widehat{d}_i = \{j | \widehat{s}_{i,j} - c_{i,j} > 1/2\}$ that is a lower bound of d_i
5) Based on $\widehat{d}_i$, one can eliminate some $X_1, \cdots, X_K$ form sampling based on similar rules like in the value based case (see section II-A)

[2]It is clear that $P(X_i \prec X_j) \approx \widehat{s}_{i,j}$ based on finite sample sets.

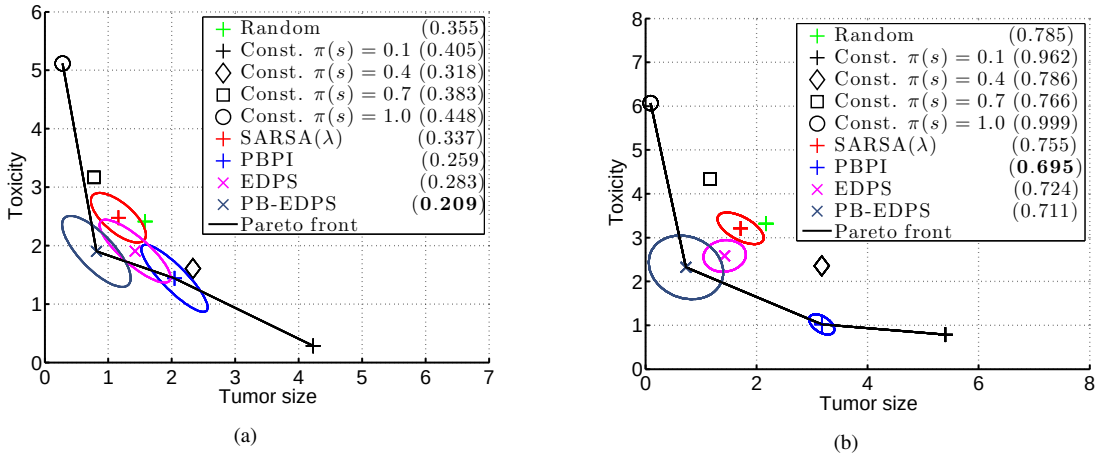

Fig. 1. Illustration of patient status under different treatment policies. On the x-axis is the tumor size after 6 and 12 months, on the y-axis the highest toxicity during the treatment. The death rates are shown in parentheses at the upper right corner.

6) If the number of iterations is bigger than $n_{\max}$ then stop and return the current approximation of Copeland's ranking based on $\widehat{d}_1, \cdots, \widehat{d}_K$.

One can show that the PBR algorithm returns the best μ random variables with respect to the surrogate decision model $\ll_C$ with high probability if $n_{\max}$ 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 π mapping states $\boldsymbol{s} = (S, X) \in \mathcal{S} = \mathbb{R}^2_+$, where S is the tumor size and 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: $\mathbf{h}' \preceq \mathbf{h}$ if the patient survives in $\mathbf{h}$ but not in $\mathbf{h}'$, and both histories are incomparable ($\mathbf{h}' \perp \mathbf{h}$) if the patient does neither survive in $\mathbf{h}'$ nor in $\mathbf{h}$. 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 $\mathbf{h}$ and $\mathbf{h}'$, 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: $\mathbf{h}' \preceq \mathbf{h}$ 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 δ and the maximum number of samples allowed for a single option, $n_{\max}$. We optimized δ in the range $\{0.01, 0.05, 0.1\}$, while $n_{\max}$ 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 $\pi_\Theta(\mathbf{s}) = 1/(1 + \exp(-\Theta^T \mathbf{s}))$. As baseline methods, we run the discrete uniform random policy (randomly choosing a dosage $d \in D' = \{0.1, 0.4, 0.7, 1.0\}$ 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 SARSA(λ) [18] with discrete action set according to the original setup[3]. 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

[3]We used an ϵ-greedy policy for exploration. Initially, the learning rate α, the exploration term ϵ and the parameter of the replacing traces λ were set to 0.1, 0.2 and 0.95 respectively, and decreased gradually with a decay factor $1/\lceil \frac{10}{\tau} \rceil$, where τ 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] 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

[1] J. Fürnkranz and E. Hüllermeier, editors. *Preference Learning*. Springer-Verlag, 2011.

[2] R. Akrour, M. Schoenauer, and M. Sebag. Preference-based policy learning. In *Proceedings ECMLPKDD 2011, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 12–27, 2011.

[3] W. Cheng, J. Fürnkranz, E. Hüllermeier, and S.H. Park. Preference-based policy iteration: Leveraging preference learning for reinforcement learning. In *Proceedings ECMLPKDD 2011, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 414–429, 2011.

[4] M. Lagoudakis and R. Parr. Reinforcement learning as classification: Leveraging modern classifiers. In *Proceedings of the 20th International Conference on Machine Learning*, pages 424–431, 2003.

[5] V. Heidrich-Meisner and C. Igel. Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In *Proceedings of the 26th International Conference on Machine Learning*, pages 401–408, 2009.

[6] R. Akrour, M. Schoenauer, and M. Sebag. April: Active preference-learning based reinforcement learning. In *Proceedings ECMLPKDD 2012, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 116–131, 2012.

[7] N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In *Parallel Problem Solving from Nature-PPSN VIII*, pages 282–291, 2004.

[8] H.G. Beyer and H.P. Schwefel. Evolution strategies–a comprehensive introduction. *Natural computing*, 1:3–52, 2002.

[9] O. Maron and A.W. Moore. Hoeffding races: accelerating model selection search for classification and function approximation. In *Advances in Neural Information Processing Systems*, pages 59–66, 1994.

[10] O. Maron and A.W. Moore. The racing algorithm: Model selection for lazy learners. *Artificial Intelligence Review*, 5(1):193–225, 1997.

[11] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

[12] J. Fürnkranz, E. Hüllermeier, W. Cheng, and S. Park. Preference-based reinforcement learning: a formal framework and a policy iteration algorithm. *Machine Learning*, 89(1-2):123–156, 2012.

[13] P. Fishburn. Nontransitive measurable utility. *J. Math. Psychology*, 26:31–67, 1982.

[14] H. Moulin. *Axioms of cooperative decision making*. Cambridge University Press, 1988.

[15] E. Even-Dar, S. Mannor, and Y. Mansour. PAC bounds for multi-armed bandit and markov decision processes. In *Proceedings of the 15th Annual Conference on Computational Learning Theory*, pages 255–270, 2002.

[16] Y. Zhao, M.R. Kosorok, and D. Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in Medicine*, 28(26):3294–3315, 2009.

[17] V. Heidrich-Meisner and C. Igel. Variable metric reinforcement learning methods applied to the noisy mountain car problem. *Recent Advances in Reinforcement Learning*, pages 136–150, 2008.

[18] G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Cambridge University, Engineering Department, 1994.

[19] Cs. Szepesvári. *Algorithms for reinforcement learning*. Morgan and Claypool, 2010.

[20] C.A.C. Coello, G.B. Lamont, and D.A. Van Veldhuizen. *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007.