International Journal of Complex Systems – Computing, Sensing and Control
Vol. 1, No. 1-2, pp. 107-117, 2013

RESEARCH OF AN OFF-POLICY BF OPTIMIZATION ALGORITHM BASED ON CROSS-ENTROPY METHOD

YUCHEN FU
Department of Information Technology
Suzhou Industrial Park Institute of Services Outsourcing
No. 99 Ruoshui Road, Suzhou, China
fuyc@siso.edu.cn

XUWEN SONG
School of Computer Science and Technology
Soochow University
No. 1 Shizi Road, Suzhou, China
songxuwen@gmail.com

ABSTRACT—In the reinforcement leaning task, the off-policy algorithms that approximately evaluate the values of states faced with the problem of high evaluation error and were sensitive to the distribution of behavior policy. In order to solve these problems, the basis function optimization method under the off-policy scenario was proposed. The algorithm set the Bellman error of the target policy which was computed with off-policy prediction algorithms as the objective function, then adjust the placement and shape of the basis functions in cooperate with the method of cross-entropy optimization. The experimental results on the grid world show that the algorithm effectively reduced the evaluation error and improved the approximation. Additionally, the algorithm could be easily extended to the problems of large state spaces.

Key Words: off-policy learning; basis function optimization; cross-entropy optimization

1. INTRODUCTION

Reinforcement Learning is an approach of learning directly from interaction to achieve a goal. The learner and decision maker is called the agent. The thing it interacts with, comprising everything outside the agent, is called the environment. Reinforcement learning framework is applicable to any goal-directed learning problem 1.

The reinforcement learning problem is consisting of prediction and control. The prediction problem that is called policy evaluation in the Dynamic Programming literature is about how to computer the state–value functions for an arbitrary policy. The control problem refers to seeking for the optimal policies which maximize the long term reward. According to whether the agent evaluates the policy that generates interactive samples or not, the prediction problem fall into two classes: on-policy and off-policy. In on-policy prediction method the behavior policy which produces samples is the same as the target policy which the agent attempt to evaluate, in other words, the distribution of the two policies are completely consistent. On the contrary, in off-policy

method the target policy is different from the behavior policy, the distribution of them may be similar or totally different. Off-policy prediction methods free the behavior policy from the target policy, enable a variety of exploration strategies to be used and permit learning from data previously collected or generated by other agents. Additionally, off-policy learning is more suitable for parallel reinforcement learning and parameter studies than on-policy learning 2.

When the state space is large or continuous, the classical reinforcement learning algorithms suffer from the "curse-of-dimensionality", thus exact solutions can't be found in general. In order to solve this problem we resort to approximation techniques like the function approximation. Apparently, the design of basis functions (BFs) plays an important role in the approximation performance regardless of which architecture of function approximation being used. Since the absence of prior knowledge which can guide the manual design of basis functions, the alternative that devises a method to automatically find BFs suited to the problem at hand is very meaningful. Two major categories of methods to find BFs automatically are BF optimization and BF construction. BF optimization methods search for the best placement and shape of a fixed number of BFs while BF construction methods add new or remove old BFs 8.

Much work has been devoted to the topic of BF optimization. The method proposed by Singh (1995) 9 was a heuristic adaptive state aggregation algorithm which pursued minimum squared Bellman error by adopted gradient descent in clustering probability space. Mannor et al. (2005) 10 proposed to adapt the basis functions and the parameters vector alternatively, combine either gradient descent or the cross-entropy method with the evaluation algorithm of least-squares temporal difference (LSTD). Bertsekas and Yu (2009) 11 gave a general framework for gradient-based BF optimization in approximate policy evaluation, and provided an efficient recursive procedure to estimate the gradient. Lucian et al. (2011) 12 introduced BF optimization into parameterized policy search, the resulting algorithm for cross-entropy policy search with adaptive BFs outperformed ordinary methods of policy search.

In spite of this existed achievements have provided some instruction on the BF optimization on the evaluation problem, it is worthwhile to study further the influence of the specific issues of off-policy prediction algorithms on BF optimization. This paper proposes an algorithm for BF optimization in off-policy prediction method which based on cross-entropy optimization. The optimization criterion is the Bellman error of target policy which calculated with data generated by behavior policy, basis function parameter and value approximation parameter are adjusted in turn as usual. The algorithm is sufficiently evaluated in the prediction problem of grid world.

2. PRELIMINARIES

2.1 Off-Policy Prediction

Reinforcement learning task always modeled as Markov decision process (MDP). An MDP is a tuple $\langle \boldsymbol{S}, \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{R} \rangle$ where $\boldsymbol{S}$ is the space of possible states of the environment, $\boldsymbol{A}$ is a set of actions available to the agent, $P : \boldsymbol{S} \times \boldsymbol{A} \times \boldsymbol{S} \to [0,1]$ defines a conditional probability distribution over state transitions given an action, and $R : \boldsymbol{S} \times \boldsymbol{A} \times \boldsymbol{S} \to \mathbb{R}$ is a reward function assigning immediate rewards to transitions. A stationary policy $\pi : \boldsymbol{S} \times \boldsymbol{A} \to [0,1]$ defines the probability of selecting each action in each state. At each discrete time step t = 0, 1, 2. . . , the agent observes the state $s_t \in \boldsymbol{S}$, and takes an action $a_t \in \boldsymbol{A}$ according to a policy π. As a result, the agent moves to a new state $s_{t+1} = s'$ with probability $P(s' \mid s_t, a_t)$ and obtains a reward $r_t = R(s_t, a_t, s')$. The prediction problem is about estimate how good it is for the agent to reach the goal in a given state and follow a particular policy. The criterion is formulated in terms of Bellman equation:

$$V^{\pi}(\mathrm{s}) = \sum_{a} \pi(s,a) \sum_{s'} P(s' \mid s_t, a_t) \left[R(s_t, a_t, s') + \gamma V^{\pi}(s') \right] \tag{1}$$

where $0 < \gamma < 1$ is the discount factor, the value function V_{π} for such a policy is the unique solution of Bellman's equation 1.

A direct solution of Bellman equation may not be feasible, either due to unknown environment dynamics (P and R) or due to the large cardinality of the state space. Therefore, methods of learning from experience and approximation or generalization technology are employed in reinforcement learning. Generally, policy evaluation is the foundation of policy improvement, thus the prediction methods should estimate approximately the whole state space as far as possible, and that means prediction algorithm must be accompanied with exploration mechanism. Off-policy methods keep explore state space while evaluating arbitrary policy by the use of soft behavior policy, different from the on-policy methods which could only evaluate stochastic policy subject to the problem of maintaining exploration.

The primary difficulty of off-policy prediction is that there is a mismatch of distributions—we would like data drawn from the distribution of the target policy but only data drawn from the distribution of the behavior policy is available. Importance sampling is a classical technique for handling this kind of mismatch which estimating the expected value of a random variable with a certain distribution from samples drawn from another distribution. Its classic form is:

$$E_{d_1}\{x\} = \int x d_1(x) \mathrm{d}x = \int x \frac{d_1(x)}{d_2(x)} d_2(x) \mathrm{d}x = E_{d_2}\{x \frac{d_1(x)}{d_2(x)}\} \tag{2}$$

Every sample is weighted by the ratio of its likelihood of occurring under the two distributions to obtain a consistent approximation of the expectation.

Importance sampling can be applied flexibility to episode or each decision step, the former constitute off-policy Monte Carlo evaluation algorithm, and the later accompanied with eligibility trace is applicable to many temporal difference algorithms to achieve off-policy evaluation.

There exist several stable algorithms for off-policy prediction such as residual gradient algorithm, LSTD, the family of gradient descent based temporal difference algorithm. Among the GQ(λ) 6 which adopt linear function approximation is an off-policy algorithm of *O(n)* complexity, that allow for use on large scale real time application. It aims to minimize the mean square projected Bellman error by gradient descent technique and the update rules are:

$$\theta_{t+1} = \theta_t + \alpha_{\theta,t} [\delta_t e_t - \gamma(1-\lambda_t)(w_t^{\mathrm{T}} e_t) \bar{\phi}_{t+1}] \tag{3}$$

$$w_{t+1} = w_t + \alpha_{w,t} [\delta_t e_t - (w_t^{\mathrm{T}} \phi_t) \phi_t] \tag{4}$$

$$e_t = \phi_t + \gamma \lambda_t \rho_t e_{t-1} \tag{5}$$

where

$$\delta_t = r_{t+1} + \gamma_{t+1} \theta_t^{\mathrm{T}} \bar{\phi}_{t+1} - \theta_t^{\mathrm{T}} \phi_t \tag{6}$$

$$\bar{\phi}_t = \sum_{a} \pi(s_t, a) \phi(s_t, a) \tag{7}$$

$$\rho_t = \frac{\pi(s_t, a_t)}{b(s_t, a_t)} \tag{8}$$

$\alpha_{\theta,t}$ and $\alpha_{w,t}$ are constant or decreasing step size parameters for $\boldsymbol{\theta}$ and $\boldsymbol{w}$ weights respectively, γ is discounted factor is parameter for eligibility trace, $\boldsymbol{\delta}$ is TD-error.

2.2 Cross-Entropy Optimization

Consider the following minimization problem 8:

$$\min_{\xi \in \Xi} F(\xi) \tag{9}$$

where $F : \Xi \to \mathbb{R}$ is the objective function to minimize, and the variable ξ takes values in the domain Ξ.Let the maximum of F be denoted by F^*.

The cross-entropy (CE) method for optimization maintains a density which has some parametric form with support Ξ. In each iteration, several samples of $\boldsymbol{\xi}$ are drawn from this density and the score values for these samples are computed. Only a few samples that have the smallest scores are kept, and the remaining samples are discarded. The parameter of probability density which we call meta–parameter is then updated using the reserved samples, so that in the next iteration the probability of generating better samples is increased. The algorithm stops when the score of the worst selected sample no longer improves significantly.

We assume that the family of densities with support Ξ and parameterized by $\boldsymbol{v}$ is $\{p(\cdot; v)\}$. At an iteration of the CE optimization algorithm, a number of N_{CE} samples are drawn from density $p(\cdot; v_{\tau-1})$; their objective values are calculated and sorted increasingly. Set $\rho_{CE} \in (0,1)$ as retention rate of samples, then the $(1-\rho_{CE})$ quantile δ_τ is determined, that is $\delta_\tau = F_{\lceil (1-\rho_{CE}) N_{CE} \rceil}$. Samples in front of δ_τ are reserved and the left are abandoned, define an indicator function:

$$I(F(\xi_{\tau_i}) \le \delta_\tau) = \begin{cases} 1 & F(\xi_{\tau_i}) \le \delta_\tau \\ 0 & F(\xi_{\tau_i}) > \delta_\tau \end{cases} \tag{10}$$

Then an associated stochastic problem is defined, which involves estimating the probability that the objective value of a sample drawn from $p(\cdot; v_{\tau-1})$ is at least δ_τ :

$$\mathrm{P}_{\xi \sim p(\cdot; v_{\tau-1})}(F(\xi) \le \delta_\tau) = \mathrm{E}_{\xi \sim p(\cdot; v_{\tau-1})}\{I(F(\xi) \le \delta_\tau)\} \tag{11}$$

The probability above can be estimated by importance sampling. For this problem, an importance sampling densities is one that increases the probability of rare event $F(\xi) \le \delta_\tau$. An optimal importance sampling density in the family $\{p(\cdot; v)\}$, in the smallest cross-entropy sense, is given by a solution of:

$$\arg\max_{v} \mathrm{E}_{\xi \sim p(\cdot; v_{\tau-1})}\{I(F(\xi) \le \delta_\tau) \ln p(\xi; v)\} \tag{12}$$

An approximate solution v_τ is computed using:

$$v_\tau = v_\tau^{'}, \text{ where } v_\tau^{'} \in \arg\max_{v} \frac{1}{N_{CE}} \sum_{\tau_i=1}^{N_{CE}} I(F(\xi_{\tau_i}) \le \delta_\tau) \ln p(\xi_{\tau_i}; v) \tag{13}$$

instead calculate the probability directly. The CE optimization proceeds with the next iteration using the new density parameter v_τ and stop when satisfy some condition. The goal of optimization is to eventually converge to a density that generates samples close to optimal values of $\boldsymbol{\xi}$ with very high probability.

3. THE BF OPTIMIZATION ALGORITHM IN OFF-POLICY SCENARIO

We choose the linear function approximate to approximate the state values, the formula is:

$$V_{\theta^\pi}(s) = f_s^T \theta^\pi = \sum_{i=1}^{n} f_s(i)\theta^\pi(i) \tag{14}$$

where θ^π is an n-dimensional parameter vector, $\phi_s = (\phi_s(1), \phi_s(2), ..., \phi_s(n))^T$ is the features vector or basis functions vector of state s which encoded by Gaussian radial basis function (GRBF):

$$\phi_s(i) = e^{(-\frac{\|s-c_i\|^2}{2\sigma_i^2})} \tag{15}$$

Every GRBF is parameterized by the core c_i and width σ_i, so the CE optimization is used to search for an optimal parameter vector $\xi = [c_1, \sigma_1, c_2, \sigma_2, \ldots c_n, \sigma_n]^T$.

We'd better let $p(\cdot; v)$ belong to the natural exponential family to simplify computation. For instance, when $\{p(\cdot; v)\}$ is the family of Gaussians parameterized by the mean μ and the standard deviation η, then $v = [\mu_{c_1}, \eta_{c_1}, \mu_{\sigma_1}, \eta_{\sigma_1} \ldots \mu_{c_n}, \eta_{c_n}, \mu_{\sigma_n}, \eta_{\sigma_n}]^{\mathrm{T}}$ and the dimension is $4n$, so the elements of $\boldsymbol{v}$ are updated as:

$$\mu_\tau = \frac{\sum_{i_s=1}^{N_{CE}} I(F(\xi_{i_s}) \le \delta_\tau)\xi_{i_s}}{\sum_{i_s=1}^{N_{CE}} I(F(\xi_{i_s}) \le \delta_\tau)} \tag{16}$$

$$\eta_\tau = \sqrt{\frac{\sum_{i_s=1}^{N_{CE}} I(F(\xi_{i_s}) \le \delta_\tau)(\xi_{i_s} - \mu_\tau)^2}{\sum_{i_s=1}^{N_{CE}} I(F(\xi_{i_s}) \le \delta_\tau)}} \tag{17}$$

The choice of objective function affects the quality of BF optimization. The best criterion is the mean squared error (MSE) between the true value and approximation derived from off-policy algorithm:

$$F_{MSE}(\theta^\pi) = \sum_{s \in RS} w(s)(\widehat{V^b_{\theta^\pi}}(s) - V^\pi(s))^2 \tag{18}$$

where ***RS*** is a representative subset of state space and *w(s)* is weight for state s which means the importance in ***RS***. We generally set $w(s) = \frac{1}{|\boldsymbol{RS}|}$, represent that the states in representative subset are equally important. V^π stands for the true value of policy π, and $\widehat{V^b_{\theta^\pi}}$ stands for the off-policy approximation which utilizes linear function approximator and behavior policy b.

As the true value is unavailable in model-free problem or large state space task, we resort to some alternatives. One is substitute Monte Carlo estimator $\widetilde{V}^\pi$ for V^π, and then the objective function is:

$$F_{MSE}(\theta^\pi) = \sum_{s \in RS} w(s)(\widehat{V^b_{\theta^\pi}}(s) - \widetilde{V}^\pi(s))^2 \tag{19}$$

Intuitively, it needs to estimate all the states in ***RS*** using direct simulation. The other one is adopt mean squared Bellman error (MSBE) which is the difference between the two sides of the Bellman equation:

$$F_{MSBE}(\theta^{\pi}) = \sum_{s \in RS} w(s)(\widehat{V^{b}_{\theta^{\pi}}}(s) - \sum_{a} \pi(s,a) \sum_{s'} P(s' \mid s,a)[R(s,a,s') + \gamma \widehat{V^{b}_{\theta^{\pi}}}(s')])^2 \tag{20}$$

where the transition probability P and expected reward R are unknown. But we can estimate them statistically from samples generated by behavior policy.

When the environment is a deterministic MDP, the expected reward needn't be estimate, and objective function can be simplified as:

$$F_{MSBE}(\theta^{\pi}) = \sum_{s \in RS} w(s)(\widehat{V^{b}_{\theta^{\pi}}}(s) - \sum_{a} \pi(s,a)[r(s,a) + \gamma \widehat{V^{b}_{\theta^{\pi}}}(s')])^2 \tag{21}$$

When the environment is stochastic MDP, let $\widetilde{P}(s' \mid s,a)$ and $\widetilde{R}(s,a,s')$ stand for the probability and the averaged reward of transition from state s to s' by take action a calculated from samples respectively. Then the objective function is:

$$F_{MSBE}(\theta^{\pi}) = \sum_{s \in RS} w(s)(\widehat{V^{b}_{\theta^{\pi}}}(s) - \sum_{a} \pi(s,a) \sum_{s'} \widetilde{P}(s' \mid s,a)[\widetilde{R}(s,a,s') + \gamma \widehat{V^{b}_{\theta^{\pi}}}(s')])^2 \tag{22}$$

There are two parameters that should be optimized: the parameter $\boldsymbol{\theta_{\pi}}$ of value function approximation and the parameter $\boldsymbol{v}$ which decide the shape and location of BFs. The algorithm proceeds by interleaving optimization steps for either $\boldsymbol{\theta_{\pi}}$ or $\boldsymbol{v}$, $\boldsymbol{\theta_{\pi}}$ is optimized by GQ(λ) and the optimization of $\boldsymbol{v}$ is based on the method of cross-entropy optimization. The pseudo-code of the algorithm is given in Table 1.

While the convergence of CE optimization is not theoretically guaranteed in general, the algorithm is usually convergent in practice. As we can imagine, most of the computational load is generated by the simulations required to estimate the value of each sample. Set the number of iterations needed as l, the time needed to evaluate a policy with off-policy algorithm as $t_{off-policy}$, the sorted time of objective values sequence as t_{order}, then the required time for off-policy BF optimization algorithm is:

$$l \cdot (N_{CE} \cdot t_{off-policy} + t_{order})$$

Actually, the complexity of this algorithm is linear in the number of iterations, the number of the samples each iteration and the time required by the off-policy learning algorithm.

4. EXPERIMENTAL RESULT

Grid world is the most popular experiment used to measure the performance of prediction algorithms in reinforcement learning due to the attributes of simple, extensible and easily computed. We set a discrete two-dimensional grid world at first among which the red parts are goal states and the black part is barrier as Figure 1 depict. There are 25×25 states and the set of action is {up, down, right, left}. The state transitions are stochastic: every move will succeed with probability 0.9, stay still with probability 0.05 and move to opposed direction with probability 0.05. The agent receives a reward of -1 for each step, and a reward of 10 for reaching the goal

Table 1. Off-policy BF optimization algorithm

Input: behavior policy b, target policy π, representative states ***RS***, weight function w, GQ(λ) parameters $0<\gamma<1$, $0<\lambda<1$, $\alpha_w>0$, $\alpha_\theta>0$, CE optimization parameters $\rho_{CE} \in (0,1)$, $N_{CE} \geq 10$, $d_{CE} \geq 2$, $\tau_{\max} \geq 2$, $\varepsilon_{CE} \geq 0$.

1: $\tau \leftarrow 0$
2: initialize parameter $\boldsymbol{v}$, ensure that the whole state space be covered
3: **repeat**
4: $\tau \leftarrow \tau + 1$
5: generate samples $\xi_1, \ldots, \xi_{N_{CE}}$ from Gaussians given by $v_{\tau-1}$
6: **for** $i_s=1,\ldots,N_{CE}$ **do**
7: run GQ(λ) with BFs determined by ξ_{i_s} and obtain θ_{i_s}
8: compute the score $F(\xi_{i_s})$ with θ_{i_s}
9: **end for**
10: reorder and reindex samples increasingly according to score *s.t.* $F_1 \leq \ldots \leq F_{N_{CE}}$
11: $\delta_\tau \leftarrow F_{[(1-\rho_{CE})N_{CE}]}$
12: $i_\tau \leftarrow \lceil (1-\rho_{CE})N_{CE} \rceil$, index of the last of the best samples
13: update $\boldsymbol{v}$: $\mu_\tau \leftarrow \frac{1}{i_\tau}\sum_{i_s=1}^{i_\tau}\xi_{i_s}$, $\eta_\tau \leftarrow \sqrt{\frac{1}{i_\tau}\sum_{i_s=1}^{i_\tau}(\xi_{i_s}-\mu_\tau)^2}$
14: **until** ($\tau > d_{CE}$ and $\left|\delta_{\tau-\tau'} - \delta_{\tau-\tau'-1}\right| \leq \varepsilon_{\text{CE}}$, for $\tau' = 0,\ldots d_{CE}-1$) **or** $\tau = \tau_{\max}$

Output: best sample ξ^*, its objective value, and corresponding linear parameter θ_{ξ^*}

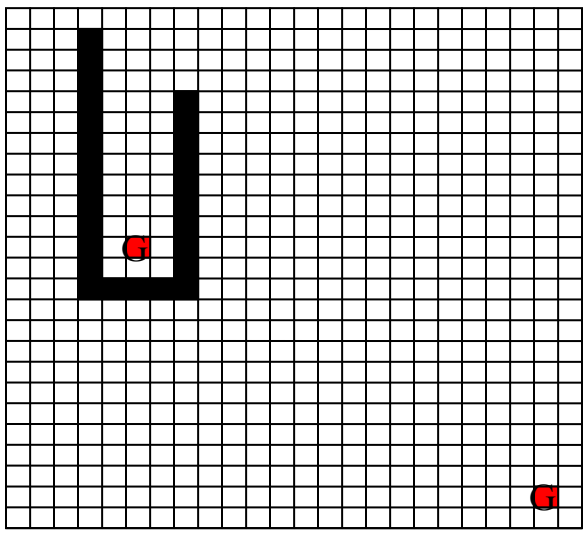

Figure 1. Grid world

states. The target policy prefers the direction down and the action probability for each state is {up: 0.1, down: 0.6, right: 0.1, left: 0.2}. Since the state space is small in this setting, we set the whole state space as representative subset and the weight for each state is equal. Additionally, the true value can be computed easily by solving Bellman equation. Objective function is formula (22).

There are three behavior policies used to compare the efficiency of BF optimization. One is a completely random policy b_1= {up: 0.25, down: 0.25, left: 0.25, right: 0.25}, one is extremely different policy b_2= {up: 0.55, down: 0.2, left: 0.15, right: 0.1}, the last one is a similar policy b_3= {up: 0.15, down: 0.6, left: 0.15, right: 0.1}. The number of BFs is 11 and the other parameters are: d_{CE}=4, ε_{CE}=0.01, τ_{max}=150, λ=0.9, γ =0.9.

The performance measure of algorithm is the MSE and MSBE, the simulation results for different value of parameters are present in Figure 2 and Figure 3.

For each behavior policy, set ρ_{CE}=0.1 and vary N_{CE} from 60 to 120 with step size 20, run the algorithm for ten times, record the objective values finally converged and plot the expectations and standard deviations of MSBE in the left of Figure 2. To illustrate actual performance of approximation, the corresponding MSE between true value and finally off-policy approximation is depicted in the right. As we can see, the curve trends of MSBE and MSE are similar for every behavior policy, conform that the MSBE as criteria is appropriate. When apply BF optimization to off-policy prediction with the behavior policy b_1, the MSBE and MSE have been sharply reduced, the performance of approximation even outperform on-policy evaluation computed with TD(λ). Moreover, the deviation is small enough to show the stability of optimization algorithm in this situation. However, when algorithm combines with behavior policy b_2 or b_3, the variance of error especially MSE is relatively large, and the actual capability of approximation is barely satisfactory in spite of the objective function has been optimized steadily. The fluctuation and poor performance of optimized results derives from samples generated by policy with preference which have been underestimated. It is an inherent weakness of off-policy algorithm in biased behavior policy situation and can't be eliminated by optimization 13. But it can be weakened somewhat, as Figure 2 depict, the averaged performance have been improved somewhat. Additionally, behavior policy b_2 is more sensitive to parameters than the others that maybe due to its potentially capability of evaluate similar policies.

Correspondingly, we depict the influence of ρ_{CE} on BF optimization in Figure 3. For each behavior policy, the final results get worse as retention rate increase. It is natural that the more samples reserved, the more information of suboptimal setting is applied to update meta-parameters. Theoretically, N_{CE} is bigger while ρ_{CE} is smaller, the better. But in practical problems the computing resources limit the growth of N_{CE}, we have to keep N_{CE} not too large to produce BF optimization algorithm of tolerable time requirement. Even if N_{CE} is small, the value of ρ_{CE} shouldn't be large; we'd better choose ρ_{CE} around 0.1.

Next we expand the grid world to continuous state space. We use the same geometric structure and environment model except uniform noise in [0.2, 0.4] is added to the transition. For example, when the agent choose action up in state (*x, y*), the subsequent state will be (*x, y+0.5+noise*) with probability 0.9 where $noise \sim U(0.2, 0.4)$. The goal is reached when the distance from the current position to the goals is less than 0.1 (using the L1-norm). The representative subset includes all integer states as the discrete grid world with $w(s) = \frac{1}{|\boldsymbol{RS}|}$. As the statistically dynamics are hardly computed and stored, objective function of Monte Carlo simulation (19) is applied. Every episode starts by state randomly selected in ***RS*** and records the complete return. The simulation processes for N_{CE} = 60 and ρ_{CE} = 0.1 are presented in Figure 4. It may be observed that the relative improvement which is monopoly and converged in the objective

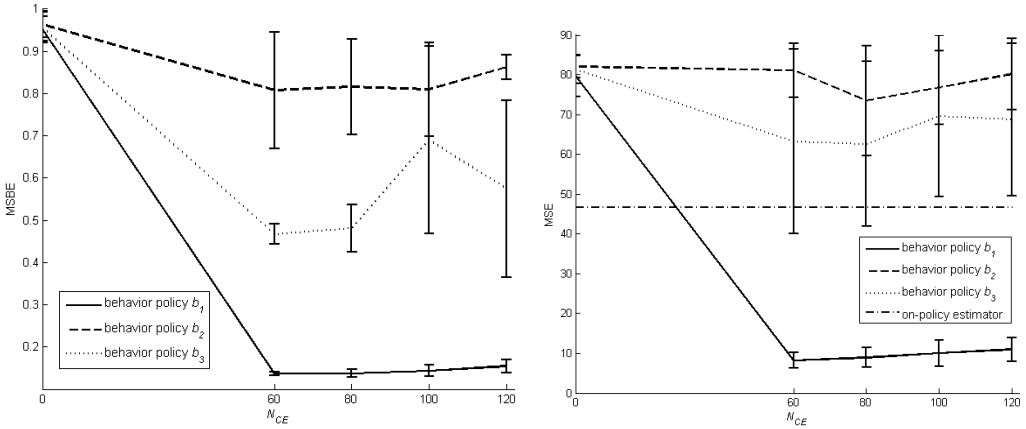

Figure 2. The effect of NCE on BF optimization performance when ρCE=0.1. The left is objective function MSBE finally achieved and the right is MSE between final approximation and true value. Results are averaged over 10 runs with an error bar representing the empirical standard deviation. The on-policy estimator for comparison is obtained by TD(λ).

function MSBE is similar to the small state space, indicate that the BF optimization algorithm in off-policy scenario is effective even in large state space. Like in discrete grid world, behavior policy b_1 with BF optimization perform best, followed by b_3, while b_2 is the worst.

5. CONCLUSION

We have addressed in this paper a basis function optimization algorithm which based on cross-entropy optimization in off-policy situation and the validity is verified by experiment. The algorithm and simulations presented here demonstrate the feasibility of basis function tuning in off-policy scenario with different behavior policy. However, the computation of BF optimization has limited its application in on-line learning. Besides, basis functions change over time, invalidate the convergence requirement of most control algorithm, that is a mainly challenge to BF optimization with control problems. An important research direction, which is an extension of the work in this paper, is to devise BF optimization methods for off-policy control. For example, combine Q-learning, the most famous off-policy learning algorithm, with BF optimization and guarantee the convergence.

6. ACKNOWLEDGEMENTS

This work was supported in part by a grant from The National Natural Science Foundation of China (61070122, 61070223, 61373094); Jiangsu Provincial Key Laboratory for Computer Information Processing Technology (kjs1024); Jiangsu Province Support Software Engineering R&D Center for Modern Information Technology Application in Enterprise (SX200902).

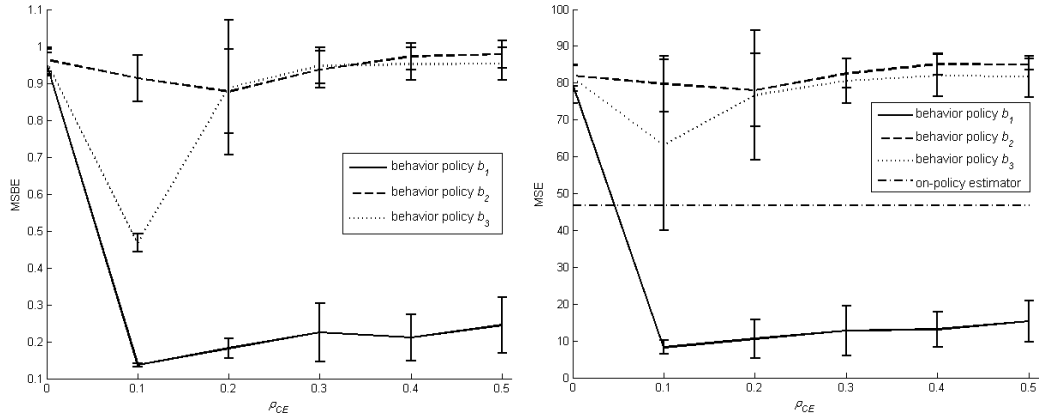

Figure 3. The effect of ρ_{CE} on the performance of BF optimization when N_{CE}=60. The left is objective function MSBE and the right is MSE between final approximation and true value. Results are averaged over 10 runs with an error bar representing the empirical standard deviation. The on-policy estimator for comparison is obtained by TD(λ).

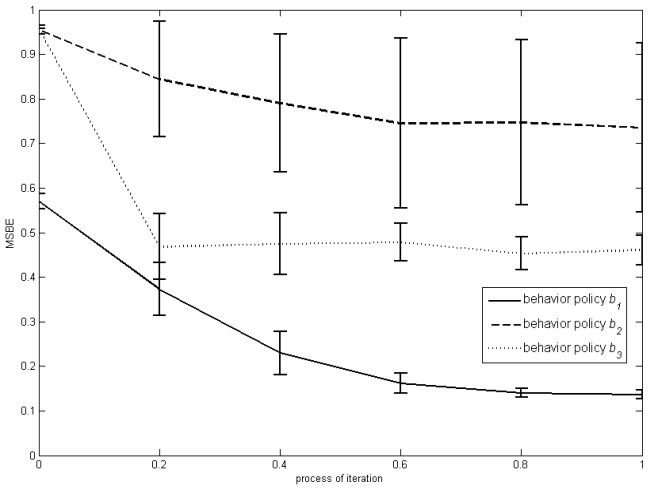

Figure 4. Performance of the off-policy BF optimization algorithm for the continuous grid world. The x-axis is stage of optimization. Results are averaged over 10 runs with an error bar representing the empirical standard deviation.

REFERENCES

1. Sutton R. S., Barto A. G. Reinforcement learning: an introduction [M]. Cambridge: MIT press, 1998.
2. Precup D, Sutton R, Dasgupta S. Off-policy temporal-difference learning with function approximation[C]//In Proceedings of the 18th International Conference on Machine Learning, California: Morgan Kaufmann, 2001: 417–424.

3. Precup D, Sutton R S, Singh S. Eligibility traces for off-policy policy evaluation [C]//Proceedings of the 17th International Conference on Machine Learning. California: Morgan Kaufmann, 2000:759–766.
4. Precup D, Paduraru C, Koop A, et al. Off-policy learning with options and recognizers[C]// Advances in Neural Information Processing Systems 18. Cambridge: MIT Press, 2005: 1097-1104.
5. Sutton R S, Szepesvari C, Maei H R. A convergent O (n) algorithm for off-policy temporal-difference learning with linear function approximation[C]//Advances in Neural Information Processing Systems 21. Cambridge: MIT Press, 2009:1609–1616.
6. Maei H R, Sutton R S. GQ (λ): A general gradient algorithm for temporal-difference prediction learning with eligibility traces[C]//Proceedings of the Third Conference on Artificial General Intelligence. Amsterdam: Atlantis Press 2010, 1: 91-96.
7. Maei H R. Gradient temporal-difference learning algorithms [D]. Canada: University of Alberta, 2011.
8. Busoniu L, Babuska R, De Schutter B, et al. Reinforcement learning and dynamic programming using function approximators [M]. Florida :CRC Press, 2010.
9. Singh S P, Jaakkola T, Jordan M I. Reinforcement learning with soft state aggregation[C]// Advances in neural information processing systems 7. Cambridge: MIT Press, 1995:361-368.
10. Menache I, Mannor S, Shimkin N. Basis function adaptation in temporal difference reinforcement learning [J]. Annals of Operations Research, 2005, 134(1): 215-238.
11. Yu H, Bertsekas D P. Basis function adaptation methods for cost approximation in MDP[C]//In proceedings 2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning. IEEE, 2009: 74-81.
12. Busoniu L, Ernst D, De Schutter B, et al. Cross-entropy optimization of control policies with adaptive basis functions [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2011, 41(1): 196-209.
13. Delp M. Experiments in off-policy reinforcement learning with the GQ(λ) algorithm [D]. Canada: University of Alberta, 2010.

ABOUT THE AUTHORS

Y. Fu received the M. Sc. degree in computer application from Wuhan University of Hydraulic and Electrical Engineering, China in 1997, and the Ph.D. degree in computer application technology from Wuhan University, China in 2003. Yuchen Fu joined the Department of computer at Wuhan University of Hydraulic and Electrical Engineering, China in 1997. Currently he is an Associate Professor and the Head of the Information Technology Department at Suzhou Industrial Park Institute of Services Outsourcing in China. His research interests include intelligent systems, complex network, machine learning, data mining and information retrieval.

X. Song received the bachelor degree in management from Henan University of Economics and laws, China in 2011. Currently she is a postgraduate majored in management science and engineering under the instruction of prof. Fu in Soochow University, China. Her research interests include reinforcement learning, approximate dynamic programming and vehicle routing problems.