Z-learning of linearly-solvable Markov Decision Processes[*]

Aleksandr Beznosikov[1]
beznosikov.an@phystech.edu
[1]Moscow Institute of Physics and Technology

We solve problems of optimal control over the ensemble of energy devices. The behavior of this ensemble can be described by a Markov Decision Process. For certain class of Markov Decision Processes (Linearly-Solvable Markov Decision Processes) which greatly simplify Reinforcement Learning. In this paper we adapt a modification (Z-learning) to the case of Markov Decision Process discussed in the context of energy systems and solve the optimal control problem by incomplete data. Comparing with standard Q-learning, show that modification of algorithm gives faster and more reliable solution.

Keywords: *Linearly-Solvable Markov Decision Processes, Z-learning, Q-learning.*

1 Introduction

In the area of power systems there is a huge demand on fast reinforcement learning algorithms. Especially they are needed in energy systems, which are very dependent on external influences and external conditions. These algorithms allow to respond to changes around the system, make decisions about changing the state of the system, and help you optimally adapt to customer requests.

For example, reinforcement learning is used for the derivation of efficient operating strategies for hybrid electric vehicles [8]; for solar microgrid energy management [7]; or for a smart energy building [5]. In [6] they use reinforcement learning for system with wind generator to pridict available wind power for selecting the optimal battery scheduling actions under different time-dependent environmental conditions and increasing the consumer independence from the external grid.

In this paper we solve the problem of optimal energy system [2]. Given a set of devices that can be in a certain number of states at a particular point in time. We need to transfer this system from the initial state to the optimal one. Each transition has a certain probability; being in each state has some cost; the system responds to each transition. It is necessary to minimize the waste of energy (or other resources) that we will require to transfer the system from the initial state to the final state.

The behavior of the system of devices in time is considered as a discrete Markov decision process [3]. In the general case, the solution of the discrete optimization problem is NP complete [1] and we can not solve it simply. But in [10] there is description of a specific Markov Decision Process (MDP) class – Linearly-Solvable Markov Decision Processeses, which we can solve optimally. Most methods require knowledge of what happens if a system "left alone" long enough. In practice it often happens that we don't have this information.

In this paper it is proposed to use the Z - learning method (stochastic modification of Q-learning from [4, 9]). Together with the solution of the main task of the MDP in parallel to restore unknown data on the behavior of the system. With this algorithm, a working model of system management will be built based only on limited samples of representative behavior. We compare the speed and quality of the two algorithms: Z-learning and Q-learning, when solving

[*] Supervisor: Vadim Strijov Task author: Michael Chertkov Consultant: Yury Maximov

the MDP, describing via transition probability matrix. Given initial state vector (probability of being in a state at time zero), we generate data for the time evolution of the state vector.

2 Problem Setup

2.1 Formal statement of the problem

As mentioned above, we work with a system of devices. Their behavior can be described by MDP:

Definition 1. *Markov decision process is* (S, P, Υ)*:*

1. *set of states* S*,*
2. $P(t)$ *is the transition probability matrix. The matrix element* $p_{ij}(t)$ *is the probability for a device which was at state* j *at time* t *to transition to state* i *at* $t+1$*. In this definition matrix* P *is variable. It replaces "actions" in the classic understanding of MDP*
3. $\Upsilon(t)$ *is the cost matrix. The matrix element* $\gamma_{ij}(t)$ *is encouraging for the transition from* j *to state* i*.*

This is the classic definition of MDP. In this paper and in [2] we work with other definition of MDP. There is no set of actions A, but P is variable. We turn to a similar model of the MDP. In this model deffirent matrixes P correspond to different actions in the classical understanding of the MDP.

Other notations:

- vector $\rho(t)$, where $\rho_i(t) \geqslant 0$ is the probability to find a device in state i at time t. The vector is normalized at all times ($\sum\limits_i \rho_i(t) = 1, \quad \forall t, \quad \forall j$).
- vector $u(t)$, where $u_i(t)$ is the cost for being in state i at time t.

The task formulated in [2] is:

$$\min_{p,\rho} \sum_{t=0}^{T-1} \sum_j \rho_j(t) \left(\sum_i p_{ij}(t) \left(U_i(t+1) + \gamma_{ij} \log \frac{p_{ij}}{\bar{p}_{ij}} \right) \right) \tag{1a}$$

$$\text{s.t.} \quad \sum_i p_{ij}(t) = 1, \quad \forall t, \quad \forall j \tag{1b}$$

$$\rho_i(t+1) = \sum_j p_{ij}(t) \rho_j(t) = 1, \quad \forall t, \quad \forall i \tag{1c}$$

$$\text{initial conditions: } \rho_i(0), \quad \forall i \tag{1d}$$

Here (1a) is target "cost vs welfare", (1b) - (1c) are stochastic restrictions.

P is the optimization/control variable. $\bar{P}$ is stochastic matrix describing the transition probabilities corresponding to normal dynamics within the ensemble ($\bar{P}$ explains the dynamics that the system would show if $u = 0$).

In practice, we do not have access to knowledge about the system, so methods of Q-learning and Z-learning will be suitable for solving our problem.

2.2 Machine learning statement of the problem

Let's give definitions for Q-learning and Z-learning:

Definition 2. *Reward function is a function showing rewards for $T-1$ steps:*

$$R_t = \sum_{t=0}^{T-1} r_t \tag{2}$$

Hereinafter r_t is charge at time t.

Definition 3. *Optimal value function is an expected total the profit that an agent will receive if he starts with this states and will follow the optimal strategy:*

$$V(i) = \max \left[\mathbb{E} \left(\sum_{t=0}^{T-1} r_t \right) |s_0 = i \right] \tag{3}$$

hereinafter $\mathbb{E}$ is an expected value.

Z-function is for Z-learning:

$$Z(i) = -\log(V(i)) \tag{4}$$

We works with an obvious stochastic approximation to the function Z from [10]. In the notation of our model:

We have a set of datas

$$\mathfrak{G} = (i_k, j_k, u_k)$$

where i_k is a current state, j_k - the next state, u_k - cost for being in i_k, $n = |\mathfrak{G}|$ - sample size.

$$Z(i_k) := (1-\alpha_k)Z(i_k) + \alpha_k \exp(-u_k)Z(j_k) \tag{5}$$

Algorithm 1 Z - learning

```
Initialize: $Z$
Input: $\mathfrak{G} = (i_k, j_k, u_k)$
for $k = 0, 1, 2, \ldots, n-1$
    $Z(s_k) := (1-\alpha_k)Z(i_k) + \alpha_k \exp(-u_k)Z(j_k)$
return $Z$
```

Definition 4. *Optimal action-value function is an expected total the profit that an agent will receive if he starts with this states and chooses this action and will follow the optimal strategy:*

$$Q(i,p) = \max \left[\mathbb{E} \left(\sum_{t=0}^{T-1} r_t \right) |s_0 = i, \quad a_0 = p \right] \tag{6}$$

We have a set of datas

$$\mathfrak{G} = (i_k, j_k, l_k, p_k)$$

where $n = |\mathfrak{G}|$ - sample size, i_k is a current state, j_k - the next state, l_k isn't cost for being in i_k, but a total cost, which depends on u and Υ. The equation for Q-learning is

$$Q(i_k, p_k) := (1 - \alpha_k) Q(i_k, p_k) + \alpha_k \min_{p'} (l_k + Q(j_k, p')) \tag{7}$$

Algorithm 2 Q - learning

```
**Initialize:** $Q$
**Input:** $\mathfrak{G} = (i_k, j_k, l_k, p_k)$
**for** $k = 0, 1, 2, \ldots, n-1$
    $Q(i_k, p_k) := (1 - \alpha_k) Q(i_k, p_k) + \alpha_k \min_{p'} (l_k + Q(j_k, p'))$
**return** Q
```

3 Numerical experiments

We carried out numerical experiments of two algorithms on a randomly generated sample (all transitions and all transition are random, but in one place there is a hole, so that there is an optimal path to which the algorithm should converge). We run the experiment several times, with different initial values.

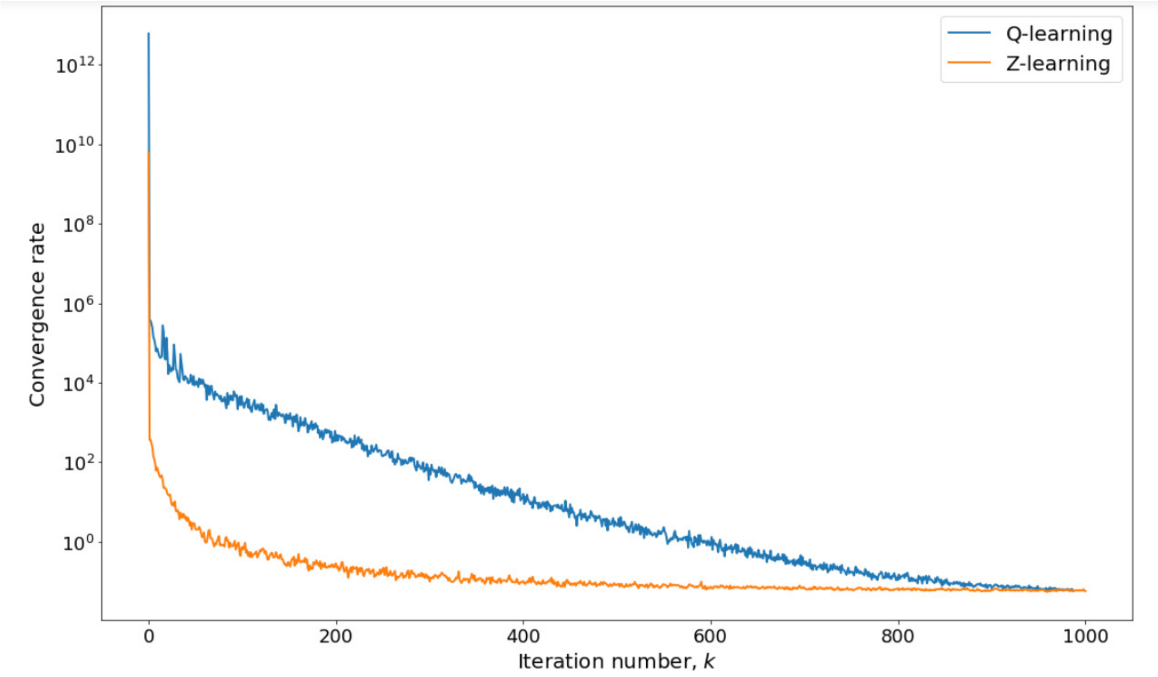

Figure 1 Numerical experiments of Z-learning and Q-learning

We obtained that on such data the Z-learning algorithm shows itself better - converges more quickly to the optimal solution.

4 Conclusion

We reviewed the work of two learning algorithms with reinforcement in relation to the task of energy ensemble. Numerical experimentation shows that the Z-learning algorithm shows itself better than the Q-learning algorithm and is a good modification that is suited to this task.

References

[1] Jens Vygen Bernhard Korte. *Combinatorial optimization. Theory and Algorithms.* MCCME, translation edition, 2015.

[2] Michael Chertkov, Vladimir Y Chernyak, and Deepjyoti Deka. Ensemble control of cycling energy loads: Markov decision approach. In *Energy Markets and Responsive Grids*, pages 363–382. Springer, 2018.

[3] Richard Durrett. *Probability: theory and examples.* Duxbury Press, Belmont, CA, second edition, 1996.

[4] Chi Jin, Zeyuan Allen-Zhu, Sébastien Bubeck, and Michael I. Jordan. Is q-learning provably efficient? *CoRR*, abs/1807.03765, 2018.

[5] Sunyong Kim and Hyuk Lim. Reinforcement learning based energy management algorithm for smart energy buildings. *Energies*, 11:2010, 08 2018.

[6] Elizaveta Kuznetsova, Yan-Fu Li, Carlos Ruiz, Enrico Zio, Keith Bell, and Graham Ault. Reinforcement learning for microgrid energy management. *Energy*, 59:133–146, 05 2013.

[7] Raju Leo, R S Milton, and S Sibi. Reinforcement learning for optimal energy management of a solar microgrid. pages 183–188, 09 2014.

[8] Roman Liessner, Christian Schroer, Ansgar Dietermann, and Bernard Bäker. Deep reinforcement learning for advanced energy management of hybrid electric vehicles. pages 61–72, 01 2018.

[9] Csaba Szepesvári. *Algorithms for Reinforcement Learning.* Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

[10] Emanuel Todorov. Linearly-solvable markov decision problems. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *NIPS*, pages 1369–1376. MIT Press, 2006.