Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186-5140
Volume 8, Number 1, pages 19–22, January 2019

Smart Grid Optimization by Deep Reinforcement Learning over Discrete and Continuous Action Space

Tomohiro Hirata[1], Dinesh Bahadur Malla[2], Katsuyoshi Sakamoto[3], Koichi Yamaguchi[3], Yoshitaka Okada[1], Tomah Sogabe[1,2,3]

[1]Research Center for Advanced Science and Technology, The University of Tokyo, 153-8904,Japan
[2]Technology Solution Group, Grid Inc., Kita Aoyama, Minato-ku, Tokyo, 107-0061, Japan
[3]Info-Powered Energy System Research Center, Department of Engineering Science,
The University of Electro-Communications, Chofu, Tokyo, 182-8585, Japan

***Abstract*- In this work, we have applied two deep reinforcement learning (DRL) algorithms designed for both discrete and continuous action space. These algorithms were well embedded in a rigorous physical model using Simscape Power Systems™ (Matlab/Simulink™ Environment) for smart grid optimization. Bechmark test were conducted by comparing the results from the MILP (Mixed-integer linear programming) and the DRL. The results showed that the agent successfully captured the energy demand and supply feature in the training data and learnt to choose behavior leading to maximize its profit.**

Keywords—deep reinforcement learning, smart grid, optimization

I INTRODUCTION

Energy grid system containing renewable energy resources(RES) such as photovoltaic energy, wind power as well as hydropower have been considered as alternative power supply configuration. It is renovating conventional grid systems, aiming at reducing the emission of CO_2 while mitigating the global warming. A decentralized energy system is more robust and resilient against the unexpected natural disasters, which are frequently occur in countries such as Japan. However, due to the intermittent nature of RES, a mismatch between electricity supply and demand is often encountered and causes instability and limit of power output. As an effective approach to these challenges, smart grid has been proposed and has shown great technological innovation towards intelligent, robust and functional power grid [1][2].

Smart grid evolves energy transmission among different sub-smart grid utilities, which finally contribute to the efficient energy management ecosystem of energy storage, energy supply, balanced load demand over large scale grid configuration. Construction of efficient smart grid system is in principle a control optimization mathematical problem. A wide range of methods have been proposed to tackle this challenge including linear and dynamic programming as well as heuristic methods such as PSO, GA, game or fuzzy theory and so on [3]. In the recent years, studies on energy optimization in smart grid has gradually shifted to agent-based machine learning method represented by state of art deep learning and deep reinforcement learning. Especially deep neural network based reinforcement learning methods are emerging and gain popularity to for smart grid application [4][5].

In this work, we focus on the following issues and tasks:

(1) Different from previous reports, we have developed our deep reinforcement learning algorithm embedded in a rigorous physical model using Simscape Power Systems™ for smart power grid optimization. All the parameters used in smart grid represents the realistic electric circuits and detailed fluctuation regarding the voltage, frequency and phase can be therefore fully revealed, which are not available in previous reports where the constructed smart grid system could not output sufficient information.

(2) For RL, model-free off-policy deep Q-learning suing Matlab™ is developed. Actor critic and DQN are suited for addressing continuous state space and discrete action space respectively. Here we have focused on the discrete action control designed for switching the grid power supply/sell and battery charge /discharge.

(3) For continuous state and continuous action space, we have self-developed a H-DDPG (hybrid-deterministic policy gradient) algorithm, in which we have hybridized the latest deep deterministic policy gradient with the deep actor-critic stochastic policy gradient.

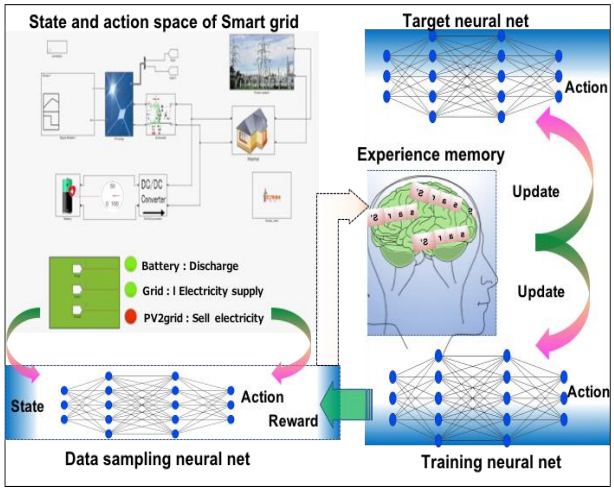

Fig.1: Sketch of smart grid optimization using deep neural network based reinforcement learning algorithm.

II ALGORITHM AND MODEL

(i) **Deep Q-Learning (DQN):** A general model which describes the main framework is given as follows. In this sketch, we adopted deep Q-learning algorithm as an example to illustrate the learning principle: physical model

Algorithm 1 Deep Q-learning with Experience Replay

Load the data from data directory for Matlab simulation
Open Simulink model
Initialize Simulink model with initial parameter {first action}
Initialize replay memory D to capacity N
Initialize weight with random weights
Initialize action with zeros (in this simulation 7 actions)
for episode = 1:M do
  Initialise state s = {*starting Value*}
  for t = 1: T do
    set starting and end simulation time for Simulink
    set update parameter
    set Return Works pace Output 'on'
    start simulation for data which are output of simulation
    simulation out = [Photovoltaic power generation, Battery State of charge(SOC)...
      Load consumption, DC power generator]
    State= simulation out
    forward state to Deep Q-learning network
    with probability select a random action a_t otherwise select $a_t = \max_a Q^*(\varphi(s_t),a;\vartheta)$
    Deep Q-learning network return action index
    Where
    1 = switch from Photovoltaic array to DC power generator
    2 = switch from Photovoltaic array to Load
    3 = switch from DC power generator to Load
    4 = switch for battery charge
    5 = switch from Photovoltaic array to DC power generator
      &switch from Photovoltaic array to Load
    6 = switch from Photovoltaic array to DC power generator
      &switch for battery charge
    7 = switch from DC power generator to Load
      &switch for battery charge
    set action in the Simulink model
    1 = "ON"
    0 = "OFF"
    simulate Simulink Model
    get simulation data from model
    set s_{t+1} = {Next *state*} from getting simulation data from model
    execute action a_t in emulator and observe reward r_t
    store transition (s_t,a_t,r_t,s_{t+1}) in D
    set $s_t = s_{t+1}$
    until s_{t+1} is terminal
    sample random minibatch of transitions (s_t,a_t,r_t,s_{t+1}) from D

$$y_j = \begin{cases} r_j & \text{reward for terminate} \\ \eta + \gamma \max_{a'} Q(\varphi_{j+1},a';\vartheta) & \text{general reward} \end{cases}$$

    Perform a gradient descent step on $(y_j - Q(\varphi_j,a_j;\vartheta))^2$ for weight update
  end for

of smart grid simulation environment based on Simscape Power Systems™ was constructed. The state space is always continuous and action space is set either discrete or continuous for off-policy Q learning and deep policy gradient algorithm respectively. A detailed operation flow is given as follows by the form of pseudo-simulation code:

(ii) **Hybrid Deep Deterministic Policy Gradient (H-DDPG)**: Deterministic policy is in theory efficient at the late stage of simulation because the policy distribution is less variant and more deterministic. Policy gradient is usually formulated as follows, where η is the policy object function; $\theta$is the function approximation parameter (in neural network, it is the weight w); s and a correspond to the state and action $Q_\pi(s,a)$ is the state-action function under certain policy $\pi(a|s,\theta_p)$ and is the :

$$\nabla\eta(\theta_p) = E\left[Q_\pi(s,a)\,\nabla_{\theta_p} log\,\pi(a|s,\theta_p)\right] \qquad (1)$$

policy distribution function. David et al. has shown that if the policy is treated as deterministic; the above equation can be reformed as: [6]

$$Q_\pi(s,a)\,\nabla_{\theta_p} log\,\pi(a|s,\theta_p) = \nabla_a Q(s,a) \qquad (2)$$

and if the action a is approximated as policy action function:

$$a = u(s|\theta^\mu) \qquad (3)$$

using the chain rule $\nabla_a(s,a)$ can be further extended as:

$$\nabla_{\theta^a} Q(s|\theta^a)\nabla_{\theta^\mu} u(s|\theta^\mu) \qquad (4)$$

and then policy parameter θ^μ is updated as the usual gradient decent:

$$\theta^\mu = \theta^\mu + \alpha\cdot\nabla_{\theta^a} Q(s|\theta^a)\nabla_{\theta^\mu} u(s|\theta^\mu) \qquad (5)$$

However, implementing the deterministic policy at the early simulation stage will inevitably cause high variance and slow convergence because the policy is far from optimal policy so the policy distribution is fairly stochastic and less deterministic with high bias. The hybridized algorithm is designed in such a way that both the advantage of deterministic and stochastic policy is assimilated thus a stable learning profile with fast convergence can be achieved.

(iii) **Neural Network Model**: In this work, we use multilayer neural network including four hidden layers to approximate the state-action value function. The activation function is fixed at hyperbolic-tangent function and epsilon-greedy algorithm is utilized to enhance the exploration in the case of DQN for discrete action and re-parameterization. These techniques were used when using H-DDPG for continuous action space.

$$\Delta\theta^\mu = \begin{cases} \alpha\cdot\nabla_{\theta^a} Q(s|\theta^a)\nabla_{\theta^\mu} u(s|\theta^\mu), & Step_t \\ \alpha\cdot\{r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\,\nabla_{\theta^\mu} u(s|\theta^\mu), & Step_{t+1} \end{cases}$$

(iv) **MILP and DRL Model:** In this work we performed benchmark test and compared the results from the MILP and DRL algorithm. Both the MILP and DRL were perfomed under the same input data including the solar power generation and electric consumption profile as well the purchase/sell price for the electricity. The soft constraints for the battery charge/discharge were also arranged the same for both methods. The inter-conversion principle between these two methods were given in Fig.2.

For MILP we divide the constraints in into two parts : Soft constraint and hard constraint. And we make soft constraint as reward for the neural network learning process and hard constraint are difficult to learn so we used them to terminate and restart the learning process.

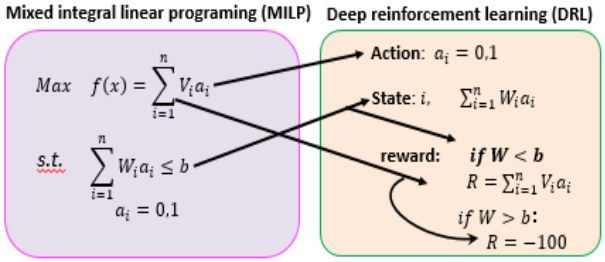

Fig.2. Conversion between MILP and DRL implementation

III RESULTS

Here we present one representative simulated results by employing DQN algorithm to optimize for discrete action space. Mainly we deploy the DRL(DQN) agent to maximize earning for comparing with MILP optimization, and we also use DRL for maintain balance from different power sources.

of problems, among them MILP is popular tools in the Matlab environment. On the base of Matlab deployed energy hub optimization we compared our result and its reliability so far. In Fig.3. upper graphs are result of Matlab based MILP used optimized result for buying and selling. The optimized one-day profit from selling power produced by PV and reduced the cost of buying from power producer is 74 yen. The lower two graphs are result by our programed DRL agent. The optimal result obtained from DRL is 78 yen for a day, from selling power produced by PV and reduced the cost of buying from power producer. By comparing these result DQN (deep reinforcement learning agent) is good enough to optimize the power system optimization problem. By using reinforcement learning agent we get optimized result as well as we can get different option for optimizing the problem where these options are not available from other optimization tools.

Optimization tools in Matlab deliver quite accurate results but failed to be applied to large scale system. The MILP results calculated over 10 days input data has greatly deviated from the theoretical solution. On the contrast, DRL has greatest advantage over the MILP in this sense. The machine learning based DRL method learn the feature of the system via big data and generalize the feature using neural network. The agent successfully learnt to discharge

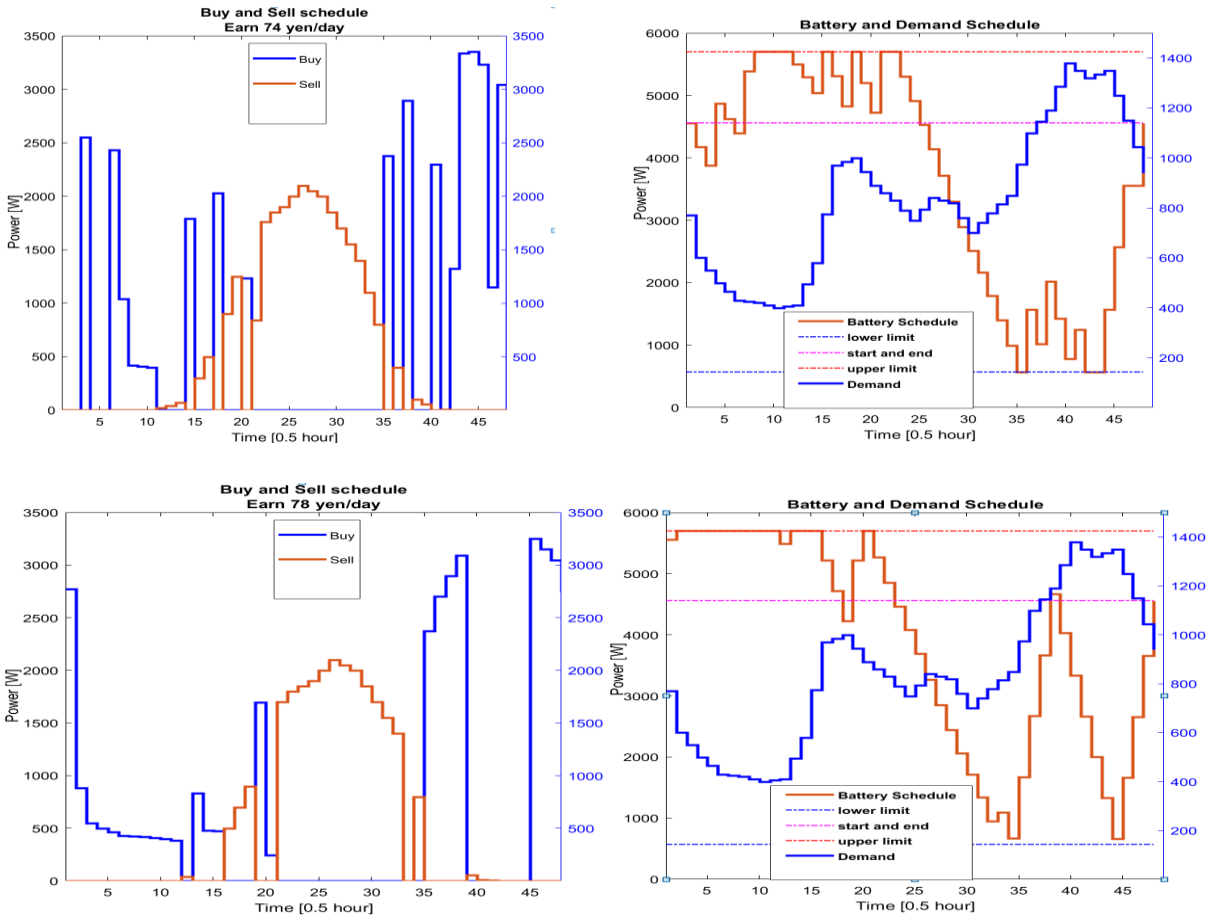

Fig.3. The MILP optimization tools optimized buying and selling schedule in the upper and Agent training results using the DQN algorithm. In the lower graph diagram

There are many optimization methods for different types its battery power during daytime instead buying electricity

from the grid and also learnt to purchase at low price time. We also compare the DRL agent and MILP optimization result on different PV production pattern and different selling buying rate. we used the same DRL reward system for all the comparing time and most of time DRL get the better result than the optimization tools.

The DRL agent is able to maintain the balance for power source demand stability. There are many options for power sources to fulfill the power demand in such problem Reinforcement agent can help to maintain supply demand balance. In Matlab environment we create virtual power resources and power demand as well, from the helps of power sources and demand data DRL agent able to maintain the power demand and supply.

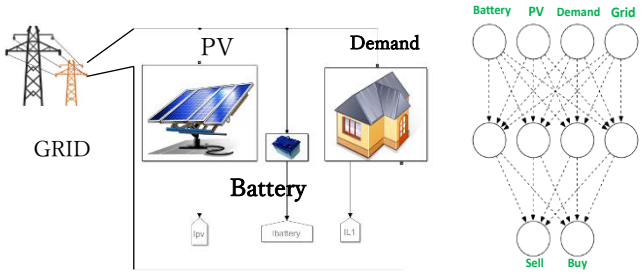

Fig.4. The model for power balance in Matlab

On the base of above works and result, we are building large scale virtual power network for demand Grid power supply as well as power purchaser many others electricity producer like PV, Turbine, Wind farm, CHP etc. and Heat producer like gas boiler, heat tank etc. as well. According to this plan we need good DRL algorithm as well as more agents. Our planed concepts simple diagram is shown in Fig.5 and more detailed design principle and preliminary results from trial experiments will be given at the conference.

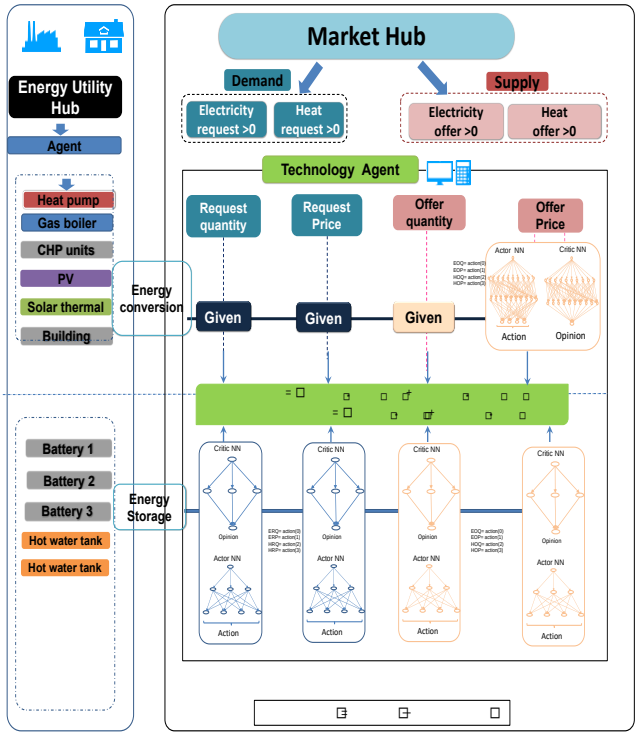

Fig.5. Sketch of large scale virtual power plant using DRL

IV CONCLUSION

We present here a deep reinforcement learning method applied for smart grid optimization. From the preliminary simulation results, the agent was able to catch the feature involved in the balance of load demand, PV power surplus and battery discharge/charge as well as grid integrate. The agent successfully learnt how to tune its action profile to maximize the reward function during training. More detailed results regarding to the comparison between DQN and H-DDPG and the key role played by reward function will be given at the conference.

REFERENCES

[1] R. H. Khan and J. Y. Khan, "A comprehensive review of the application characteristics and traffic requirements of a smart grid communications network," *Computer Networks*, vol. 57, no. 3, pp. 825-845, 2013.

[2] H. E. Brown, S. Suryanarayanan, and G. T. Heydt, "Some characteristics of emerging distribution systems considering the smart grid initiative," *The Electricity Journal*, vol. 23,no. 5, pp. 64 -75, 2010.

[3] M. R. Alam, M. St-Hilaire, and T. Kunz, "Computational methods for residential energy cost optimization in smart grids: A survey," *ACM Comput. Surv.*, vol. 49, pp. 22-34,Apr. 2016.

[4] E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Deep learning for estimating building energy consumption," *Sustainable Energy, Grids and Networks*, vol. 6, pp. 91-99, 2016.

[5] V François-Lavet, Q Gemine, D Ernst, R Fonteneau, "Towards the minimization of the levelized energy costs of microgrids using both long-term and short-term storage devices, "*Smart Grid: Networking, Data Management, and Business Models*, P295-319,2016

[6] S. David, L. Guy, H. Nicolas, D. Thomas, W. Daan, and R.Martin. "Deterministic policy gradient algorithms," *ICML*, 2014