SCIENCE CHINA
Information Sciences

• **Supplementary File** •

On Distributed Event-Based Optimization for Shared Economy in Cyber-Physical Energy Systems

Qing-Shan Jia* & Junjie Wu

Center for Intelligent and Networked Systems, Department of Automation,
Beijing National Research Center for Information Science and Technology (BNRist),
Tsinghua University, Beijing 100084, *China*

Appendix A Problem Formulation

We directly use the model in [1] and consider the problem of scheduling shared EVs. There are several independent high-rising buildings mounted with wind turbines that generate electricity. Wind power storage is not considered. So the generated wind power will not be stored or transferred to the grid and can only be used instantly. Under each building is a parking lot where the shared EVs are parked and charged. The EVs' batteries can be charged either by electricity bought from the grid market or the wind power generated from the turbines. The driving demand from users is stochastic. And the driving route is point to point, from one building to another. The goal is to maximize the global income of the shared EVs operator by properly scheduling all the EVs, including charging and picking up users. We formulate the problem into Markov decision process (MDP) [2]. The details are illustrated in the following subsections.

Appendix A.1 System State

The system state includes **three** parts, the wind power, the user demand and the state of EVs.

The charging power of EVs is assumed as fixed. Then the state of wind power can be discretized into integers, representing the number of EVs that can be charged by wind power. Denote the state of wind power as follows.

$$W_t = \{W_t^1, W_t^2, \dots, W_t^B\}. \tag{A1}$$

In (A1), B is the number of independent buildings. $W_t^i, i = 1, 2, \dots, B$ is the wind power state at the ith building at time t, the value of which can be $0, 1, 2, \dots, N_e^i$, where N_e^i is the number of EVs parked at the ith building.

State of user demand is denoted as

$$D_t = \{D_t^1, D_t^2, \dots, D_t^B\}, \tag{A2}$$

in which $D_t^i, i = 1, 2, \dots, B$ is the user demand vector at the ith building. $D_t^i = \{d_t^1, d_t^2, \dots, d_t^B\}$, in which $d_t^j, j = 1, 2, \dots, B$ represents the demand from the ith building heading for the jth building at time t.

The state of EVs is denoted as

$$V_t = (L_t^i, C_t^i, \tau_t^i), \forall i = 1, 2, \dots, N, \tag{A3}$$

in which N is the total number of EVs in the system. L_t^i is the location of the ith EV at time t. Specially, we set $L_t^i = 0$ if the ith EV is running on the road. C_t^i is the state of charge (SoC) of the ith EV. τ_t^i is the remaining driving time of the ith EV. Specially, we set $\tau_t^i = 0$ if the ith EV is parking.

Based on the above settings, the system state can be denoted as

$$S_t = (W_t, D_t, V_t). \tag{A4}$$

All the possible system states form the state space $\mathbb{S}$.

The system state in (A4) introduced in Appendix A.1 is a large high-dimensional matrix. And the state space $\mathbb{S}$ can be enormously huge as the system scale grows. Under this situation, the general optimization methods under MDP framework, like the tabular based methods, are not effective. It not only takes large storage space to store the table, but also is time-consuming to update the system state and the table. So the problem is almost intractable under this definition of system state. A simplified version of system state is needed.

* Corresponding author (email: jiaqs@tsinghua.edu.cn)

In the problem settings, each EV can be viewed as an agent. The optimization can be performed in the perspective of each EV (each agent) respectively. The goal is to find the best action for each EV under different states to maximize the objective function in (A13).

Rather than the full information about the entire system, limited local information is enough as the micro-grids (buildings) are independent from each other. For one EV, only the local information at the building where it is located matters. The state of wind power, user demand and EVs at other buildings are redundant information. Based on this idea, the system state for the ith EV at time t under distributed framework is simplified as

$$S_t^i = (W_t^{L_t^i}, D_t^{L_t^i}, V_t^j), \forall j \in \{j | L_t^j = L_t^i\}. \tag{A5}$$

The state in (A5) can be further simplified because only few information of other EVs is needed. One EV does not have to obtain the specific value of other EVs' SoCs. The key value is the ranking of SoC. We then transform the states of other EVs into one single integer, the ranking of SoC. The final version of simplified state for the ith EV at time t is denoted as

$$S_t^i = (W_t^{L_t^i}, D_t^{L_t^i}, V_t^i, R_t^i), \tag{A6}$$

in which R_t^i is the ranking of SoC of the EV. Conflicts may occur without this variable. For example, all EVs will choose to pick up an user when demands occur as they obtain substantial reward. This leads to conflicts when supply exceeds demand. The ranking of SoC helps to avoid such conflicts and we can decide which EVs are more suitable to pick up an user or to be charged.

Compared with the original state in (A4), the agent-based state in (A6) is much simpler. More importantly, each shared EV can optimize its own action respectively based on the agent-based state. Distributed optimization algorithm can be implemented. In later sections, we will explicitly introduce the Q-learning optimization method for scheduling shared EVs based on the simplified state definition in (A6).

Appendix A.2 Event

In event-based optimization, one event is a set (aggregation) of state transition pairs.

$$e_t = \{\langle S_t, S_{t+1} \rangle\}. \tag{A7}$$

In our problem, certain events like the fluctuation of the wind power, the arrival of an EV, and the change of ranking of SoC, are important information for decision making. We focus on these certain events rather than the states of the system. Then the scale of the problem is reduced. Specially, we focus on two sets of event selections, the macro-event e^M and the micro-event e^m. In our problem setting, the wind power is a macro-state component, whose state is shared by the whole system. The ranking of SoC is a micro-state, which is local information. We observe the performance under these two event selections.

$$e = \{e^M, e^m\} = \{\langle W_t, W_{t+1} \rangle, \langle R_t, R_{t+1} \rangle\}. \tag{A8}$$

The event space is denoted as $\mathbb{E}$. We define the event complexity as the number of events.

Appendix A.3 Action and Policy

The scheduling of shared EVs includes charging battery, picking up users, moving to a new building, and stay in the same state. Correspondingly, there are four options of actions for each EV, namely to charge battery, to pick up an user, to drive to another building, or to do nothing. Here the action of driving to another building means the EV is scheduled to another building without picking up an user. This action is introduced to balance the number of EVs at each building and to avoid the situation where all EVs run into one single building. And the user demand in every building can be satisfied evenly. Denote the action at time t as

$$a_t = (a_t^1, a_t^2, \ldots, a_t^N). \tag{A9}$$

in which $a_t^i, i = 1, 2, \ldots, N$ is the action for the ith EV. The value of a_t^i is denoted as $-1, 0, 1$ or 2, each representing to pick up an user, to do nothing, to charge battery, or to drive to another building without picking up an user. All the possible actions form the action space $\mathbb{A}$.

The mapping from event to action is called policy $d : \mathbb{E} \to \mathbb{A}$. A policy d can be denoted as

$$d = (a^{e^1}, a^{e^2}, \ldots, a^{e^{|\mathbb{E}|}}), \tag{A10}$$

in which $|\mathbb{E}|$ is the size of event space. The goal of the optimization problem is to find the best policy $d^\star$ to maximize the global income of shared EVs operator.

Appendix A.4 System Dynamics

In Appendix A.1, we introduced that the system state includes three parts, of which the state transitions of wind power and user demand have their own patterns of distribution. Actions will not influence the their probability of state transitions. The actions taken at every time step will only influence the state dynamics of EVs.

$$C_{t+1}^i = \begin{cases} C_t^i, & a_t^i = 0, L_t^i \neq 0; \\ C_t^i + \lambda, & a_t^i = 1, L_t^i \neq 0; \\ C_t^i - \mu, & a_t = -1 \ or \ a_t = 2 \ or \ L_t^i = 0. \end{cases} \tag{A11}$$

$$\tau_{t+1}^i = \tau_t^i - 1, when\ L_t^i = 0. \tag{A12}$$

Equation (A11) shows the dynamics of EVs' SoC, where λ and μ are the charging rate in a single time slot and the battery consumption rate while driving in a single time slot separately. $L_t^i = 0$ means the ith EV is driving on the road at time t and $L_t^i \neq 0$ otherwise. Equation (A12) shows the dynamics of remaining driving time of EVs.

Appendix A.5 Objective Function

In the context of shared EVs scheduling problem, the goal is to maximize the global income of the operator.

By satisfying the demands of users, the EVs obtain reward. We denote this reward as x_t^i. $x_t^i = \sigma$ when the ith EV drives an user to destination successfully at time t and $x_t^i = 0$ otherwise.

To maintain the running of system, EVs need to be charged when the SoC drops to a certain level. The operator need to pay the electricity bought from the grid market. Here, we assume the electricity generated from wind power is free. To minimize the charging cost, the battery charging should use the free wind power as much as possible. We denote the charging cost of the ith EV at time t as y_t^i. When charged by the electricity bought from grid market, $y_t^i = p$. The market electricity price usually fluctuates during the day. When charged by the wind power or not charged, $y_t^i = 0$.

The operator also pays for the scheduling fee when $a_t^i = 2, i = 1, 2, \ldots, N$. Denote this scheduling fee as z_t^i for the ith EV at time t. $z_t^i = h$ when $a_t^i = 2$ and $z_t^i = 0$ otherwise.

The problem we considered in this paper is formulated in a discrete time framework on a daily basis, where each day is equally discretized into 96 stages with each time slot $\Delta t = 15$ minutes. The goal is to find the policy that maximizes the daily income. Then the objective function is

$$\max_d\ J^d = \sum_{t=1}^{T} \sum_{i=1}^{N} (x_t^i - y_t^i - z_t^i). \tag{A13}$$

The one-step total reward for each EV is

$$r_t^i = (x_t^i - y_t^i - z_t^i). \tag{A14}$$

Appendix B Distributed Q-learning for EBO

We introduced the agent-based state in Appendix A.1, . The idea is based on that each EV in the system is viewed as an agent, which optimizes its policy. Though, EVs still need to know some local information to decide their actions. So information flows between EVs and operation center in the building. Figure B1 describes the information flow of the system.

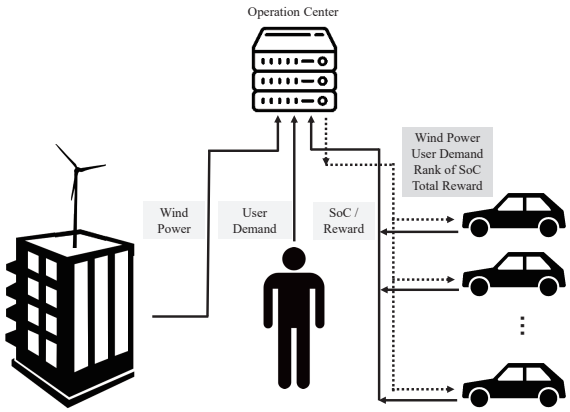

Figure B1 Information Flow in the System

At every time step, operation center in the building collects the local information including the wind power, the user demand, the state of local EVs, and the reward of each EV r_t^i. The EVs parked under the buildings send the SoCs to their local operation center, which sorts the EVs' SoCs in descending order. After that, the state of local wind power, user demand, and the ranking of SoC, the summation of the EVs' reward are sent back to each EV.

Supported by the information flow mechanism described above, we apply a distributed Q-learning based method to address the problem in (A13). Q-learning [3] is a model-free method, which means no prior knowledge about value function of the system is needed. The agent will learn the best policy on its own when the reward is properly given. (B1) is the basic function of Q-learning.

$$Q(e_t, a_t) = (1 - \alpha) \cdot Q(e_t, a_t) + \alpha \cdot (r_t + \gamma \cdot \max_a Q(e_{t+1}, a)). \tag{B1}$$

$Q(e_t, a_t)$ is the Q-factor when event e_t happened and the corresponding action is a_t. α is learning rate which determines the speed to update the current Q-factor. r_t is the one-step reward by taking action a_t when the event is e_t. γ is the discount factor which determines the weight of future rewards. $\max_a Q(e_{t+1}, a))$ is the maximum Q-factor when event is e_{t+1}.

The key issue when applying Q-learning to solve the shared EVs scheduling problem is to decide the proper reward. As introduce in Appendix A.1, the system state is based on each agent, shared EVs. Based on reward setting in Appendix A.5, they would choose to maximize the their own income, which contradicts our goal to maximize the global income of shared EVs operator. To solve this problem, we calculate the global reward at every time step and add it to each EV's reward with a weight as shown in (B2). Some other methods are studied to balance the goal. Authors in [4] established an indirect reciprocity game to achieve the cooperation between agents.

$$\widetilde{r}_t^i = (1-\omega)\cdot r_t^i + \omega \cdot \sum_{i=1}^{N} r_t^i. \tag{B2}$$

In (B2), $\omega \in [0,1]$ is the weight and $\widetilde{r}_t^i$ is weighted reward one-step reward for the ith EV.

Algorithm B1 A Q-learning Algorithm for Scheduling Shared EVs

initialize Q table with every element equaling to 0
set the upper bound of iterations I, set $l = 0$
repeat
  for each EV in the system **do**
    initialize state S^i in (A6)
  end for
  randomly select a action a^i for each EV
  for $t = 0$; $t < T$; $t++$ **do**
    for each EV **do**
      if S^i is not terminal state **then**
        update V^i according to action a^i and get $\widetilde{V}^i$
        obtain reward r^i
        if $\widetilde{L}^i > 0$ **then**
          send state of charge $\widetilde{C}^i$ and local reward r^i to operation center
        end if
      end if
    end for
    for each EV **do**
      calculate weighted one-step reward $\widetilde{r}^i$ in (B2)
      update Q according to (B1)
    end for
    if Q no longer updates **then**
      end algorithm
    end if
    for each building in the system **do**
      collect wind power W_{t+1}^j, user demand D_{t+1}^j and each EV's reward r^i
      sort state of charge of local EVs
      send W_{t+1}^j, D_{t+1}^j and ranking of SoC to each local EV and obtain $\widetilde{S}^i$
    end for
    for each EV **do**
      $S^i \leftarrow \widetilde{S}^i$
      select a action a^i according to ϵ-greedy method
    end for
  end for
  $l = l + 1$
until $l > I$

How to decide ω? The value of ω is ranging from 0 to 1. Several values (0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.5) are tested in simple cases in order to decide ω. We choose the value of $\omega = 0.1$ as it leads to better performance. The value of ω can be further fine-tuned based on this idea.

To explore the policy space, we apply ϵ-greedy method [5]. We adopt action $a^i = arg\max_a Q(e^i, a))$ with probability $1-\epsilon, \epsilon \ll 1$, and adopt $a^j \neq arg\max_a Q(e^i, a))$ with probability $\frac{\epsilon}{|\mathbb{A}|-1}$. It is a terminal state when EV runs out of battery on the road. To prevent EVs from running into terminal states, a huge penalty is given.

Based on the above analysis, we propose the algorithm for scheduling shared EVs under uncertain user demand and wind supply in **Algorithm B1**. Each EV in the system updates state and chooses action respectively. Therefore, the

computation of these processes can be allocated in each agent and distributed optimization is realized, which helps to improve the efficiency of solving the original problem.

Appendix C More Numerical Results

About the numerical experiments. A typical shared EVs system with 3 buildings (one residential building, one office building and one commercial building) and $N = 50$ shared EVs is considered. The distance between each building is set as $60km$ for simplicity. The probability of user demand heading to each type of building is shown in Figure. C1, which is based on typical routines of peoples' daily life. User demand is generated on this probability distribution and is limited to 2 user demands at most in each time step. We collect wind speed information from a weather station in Tsinghua University, Beijing, China [6]. The wind power is rescaled to fit the problem. The statistical wind power and the electricity price in market [7] are shown in Fig. C2. We assume that the wind power fluctuates with a normal distribution following the historical statistics during a day. Other system parameters are listed in Table C1. Parameters of electric vehicles are collected from BYD e6 [8].

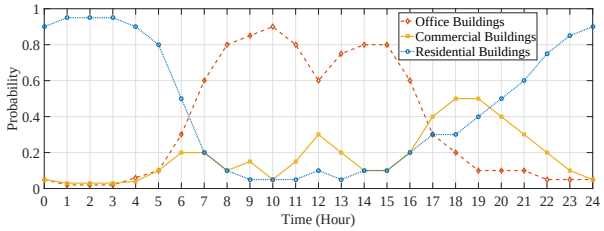

Figure C1 Arrival Probability to Each Building During the Day

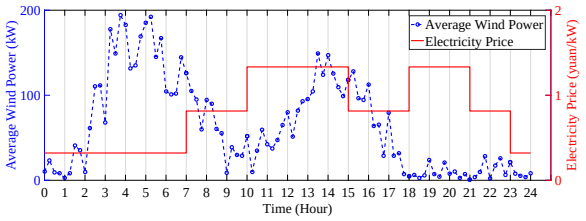

Figure C2 Wind Power and Market Electricity Price During the Day

Table C1 Parameters and values

Parameter	Value	Parameter	Value
Distance	60 km	Driving speed	50 km/h
Charging rate	5 kW	Consumption rate	5 kW
Reward σ	100	Wind power price	0
Battery capacity	60 kWh	Scheduling fee	5

We show more detailed numerical results in this section, including policy improvement rate, average time in each iteration, and the performance (objective value, wind power utilization, user demand satisfaction rate) of learned policies. We also discuss about whether to choose macro-event or micro-event under the same complexity.

Several groups of experiments under different event complexities are conducted and are listed in Table C2.

Table C2 Experiments and Event Complexities

Exp. No.	e^M Complexity	e^m Complexity
1	50	50
2	10	50
3	50	10
4	10	10
5	10	5
6	5	10
7	5	5
8	5	2
9	2	5
10	2	2
11	1	1

Appendix C.1 Policy Improvement Rate

We focus on the performance curves in Experiments 1 to 7 in the early iterations as shown in Figure C3.

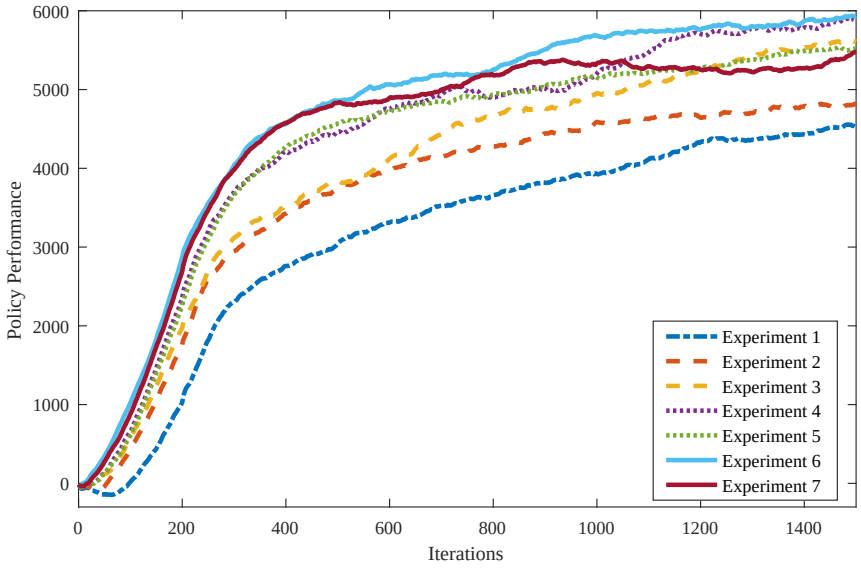

Figure C3 Policy Performance: Exp.1 - Exp.7.

The curves can be categorized into 4 groups according to the policy improvement rate.

Group 1 Experiment 1.

Group 2 Experiment 2 and 3.

Group 3 Experiment 4 and 5.

Group 4 Experiment 6 and 7.

We use different line types to represent these 4 groups of curves, which have different complexities from each other while experiments in the same group have the same complexity. From Group 1 to Group 4, the complexity decreases. Experiment 1 (with highest complexity) has the lowest policy improvement rate among others. With the complexity decreasing, the policy performance improves faster in the first 400 steps. Group 4 (Experiment 6 and 7) has the lowest complexity and at the same time the highest improvement rate. The results show that in a certain complexity range, it takes shorter iterations for the policy to converge to a satisfactory performance when the events are simpler.

Appendix C.2 Average Time in Each Iteration

We record the average time in each iteration in each experiment. The average value and the standard deviation are listed in Table C3. There are two features on average time in each iteration.

Feature 1, the average average time in each iteration drops when the event complexity decreases at first.
Feature 2, when the event complexity decreases to a certain level, it does not drop any more and even increases.

Table C3 Average Time in Each Iteration

Exp. No.	Average Time/s	Std. Deviation /s
1	1.1762	0.0480
2	0.8831	0.0610
3	0.8818	0.0760
4	0.7992	0.0620
5	0.8325	0.0468
6	0.8356	0.0605
7	0.8242	0.0440
8	0.8245	0.0352
9	0.8212	0.0350
10	0.8518	0.0451
11	0.8606	0.0411

The explanation for *Feature 1* is that when the complexity decreases, the scale of Q-table is reduced, which results in shorter time on retrieving and updating Q-table. The explanation for *Feature 2* is that when the event complexity decreases to a certain level, it takes fewer iterations to obtain a positive policy. Here the positive policy means a policy that will not lead the system to a terminal state. For example, when the EV whose SoC is low and is not sufficient to drive to the destination is scheduled to pick up a user, then it will run out of battery on the road and fail to finish the order, which case is a terminal system state. When the event complexity is low, the policy learns to avoid running into a terminal state after fewer iterations, which leads to a full length of simulation, resulting longer simulation time. While when the event complexity is high, it is more difficult to obtain a positive policy. Under a bad policy, the system runs into terminal states in the early iterations and simulations end at early stages, resulting in a shorter simulation time. Therefore, it may have longer average time in each iteration when the event complexity drops to a certain level.

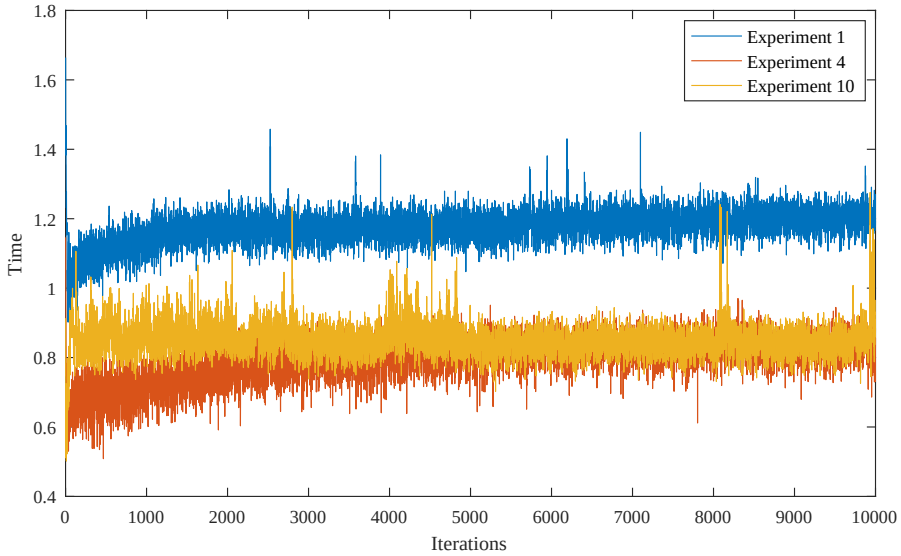

Figure C4 Time at Each Iteration Step.

Figure C4 shows the average time in each iteration in three typical experiments. Experiment 1, with the highest complexity, has longer average time in each iteration compared with the other two. In Experiment 10, with the lowest event complexity among the three, the average time in each iteration is in a stable level after only a few iterations. While in Experiment 1 and 4, it takes much more iterations before the average time in each iteration is stable at the same level. Compare Experiment 4 with Experiment 10, it is observed that after 5000 iteration steps, the average time in each iteration

in these two experiments are almost the same. The difference is in the first 5000 iteration steps, where each iteration in Experiment 4 ends in terminal state at the early stages of simulation while it does not in Experiment 10.

Appendix C.3 Macro-event or Micro-event?

Under the same event complexity, different ways of selecting events may lead to variation in performance. We observe the numerical results and compare the performance under both macro-event and micro-event when the complexity is the same. In our problem setting, the wind power is a macro-state component, whose state is shared by the whole system. The ranking of SoC is a micro-state, which is local information.

Figure C5 shows the policy performance in three groups of experiments. Experiments in each group have the same event complexity. It is observed that in each group, the policy performance under micro-event is better than macro-event. And when the event complexity decreases, the performance gap between the two increases.

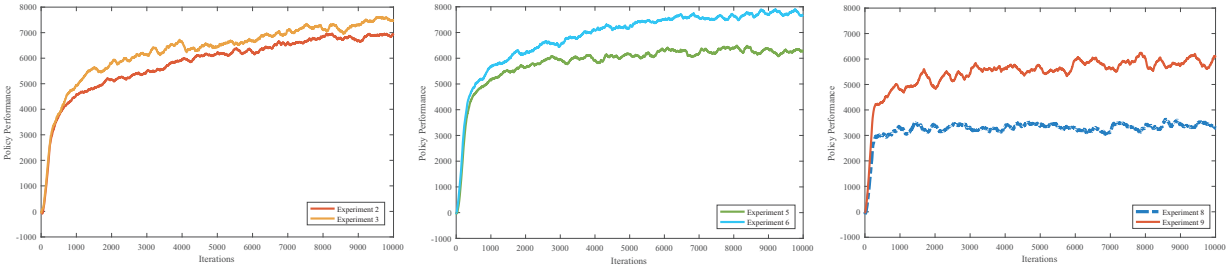

Figure C5 Policy Performance Under the Same Event Complexity.

We further study the detailed performance including objective value, wind power utilization rate (ω) and user demand satisfaction rate (μ). The performances in Experiment 5 and 6 are listed in Table C4. Note that the event complexity in Experiment 5 and 6 are the same.

Table C4 Detailed Policy Performance

Exp. No.	Exp.5 (macro-event)		Exp.6 (micro-event)	
Building	ω(%)	μ(%)	ω(%)	μ(%)
Residential	56.89($\pm$0.26)	51.66($\pm$0.89)	62.53($\pm$4.82)	70.57($\pm$8.81)
Office	47.35($\pm$1.22)	57.62($\pm$1.48)	65.35($\pm$3.02)	72.80($\pm$2.65)
Commercial	34.46($\pm$2.40)	55.01($\pm$1.44)	35.78($\pm$5.74)	71.56($\pm$8.07)
Obj. Value	6916.77($\pm$442.78)		10180.41($\pm$102.58)	

Considering the objective value, Exp.6 is almost 30% better than Exp.5. The higher objective value in Exp.6 results from the higher user demand satisfaction rate (μ). It shows that μ at each building in Exp.6 is more than 10% higher than it is in Exp.5. The higher user demand satisfaction rate leads to more income. The wind power utilization rate (ω) in the two experiments are close to each other. Exp.6 has slightly higher ω than Exp.5.

The better performance under micro-event is because the micro-event in this problem setting better captures the information and has bigger impact on the objective performance. Based on the above comparisons, it is suggested that micro-event is preferred when the event complexity is the same in the shared EVs scheduling problem.

References

1 Wu J, Jia Q S. A Q-learning Method for Scheduling Shared EVs under Uncertain User Demand and Wind Power Supply. in Proc. of 2nd IEEE Conference on Control Technology and Applications, Copenhagen, Denmark, August 2018, in press.
2 Puterman M L. Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons, Inc., 2014.
3 Watkins C, Dayan P. Q-learning. Machine Learning, vol. 8, no. 3, pp. 279-292, 1992.
4 Tang C, Li X, Wang Z, et al. Cooperation and distributed optimization for the unreliable wireless game with indirect reciprocity. SCIENCE CHINA-INFORMATION SCIENCES, 2017, 60(11): 110205.
5 Watkins C. Learning from Delayed Rewards. PhD thesis, University of Cambridge, Cambridge, United Kingdom, 1989.
6 Weather Station in Tsinghua University, China. accessed on Mar., 2018. [Online]. Available: http://climate.dest.com.cn/
7 Beijing Electricity price, accessed on Mar., 2018. [Online]. Available: http://www.bjpc.gov.cn/cxfw/jgsfcx/jzjg/201208/t9779303.htm
8 BYD e6, accessed on Mar., 2017. [Online]. Available: http://www.byd.com/na/e6/e6.html