

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier X.X/ACCESS.2017.DOI

Electric Bus Charging Schedules Relying on Real Data-Driven Targets Based on Hierarchical Deep Reinforcement Learning

JIAJU QI¹, LEI LEI¹, (Senior Member, IEEE), THORSTEINN JONSSON², AND LAJOS HANZO³, (LIFE FELLOW, IEEE)

¹School of Engineering, University of Guelph, Guelph N1G 2W1, Canada

²EthicalAI, Waterloo N2L 0C7, Canada

³School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK

CORRESPONDING AUTHOR: Lajos Hanzo (e-mail: hanzo@soton.ac.uk).

This research is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Alliance Grant (ALLRP 580528- 22)

ABSTRACT The charging scheduling problem of Electric Buses (EBs) is investigated based on Deep Reinforcement Learning (DRL). A Markov Decision Process (MDP) is conceived, where the time horizon includes multiple charging and operating periods in a day, while each period is further divided into multiple time steps. To overcome the challenge of long-range multi-phase planning with sparse reward, we conceive Hierarchical DRL (HDRL) for decoupling the original MDP into a high-level Semi-MDP (SMDP) and multiple low-level MDPs. The Hierarchical Double Deep Q-Network (HDDQN)-Hindsight Experience Replay (HER) algorithm is proposed for simultaneously solving the decision problems arising at different temporal resolutions. As a result, the high-level agent learns an effective policy for prescribing the charging targets for every charging period, while the low-level agent learns an optimal policy for setting the charging power of every time step within a single charging period, with the aim of minimizing the charging costs while meeting the charging target. It is proved that the flat policy constructed by superimposing the optimal high-level policy and the optimal low-level policy performs as well as the optimal policy of the original MDP. Since jointly learning both levels of policies is challenging due to the non-stationarity of the high-level agent and the sampling inefficiency of the low-level agent, we divide the joint learning process into two phases and exploit our new HER algorithm to manipulate the experience replay buffers for both levels of agents. Numerical experiments are performed with the aid of real-world data to evaluate the performance of the proposed algorithm.

INDEX TERMS Deep Reinforcement Learning; Electric Bus; Hierarchical Reinforcement learning; Charging Control

I. INTRODUCTION

IN recent years, the adoption of Electric Buses (EBs) in public transportation fleets has been growing rapidly. Electrification of public transport reduces GreenHouse Gas (GHG) emissions, lowers maintenance and fuel costs, and provides the public with a more comfortable riding experience [1]. With the increasing penetration of EBs, the question of how the charging cost can be reduced has become a key

concern for bus companies [2]. Concurrently, in an attempt to spread the demand for electricity and efficiently utilize the Renewable Energy Sources (RES), power utility companies have adopted real-time electricity prices. Consequently, bus companies are presented with an effective measure to reduce costs and participate in the grid services by implementing intelligent charging schedules.

In order for EBs to fully take advantage of the time-

varying electricity prices, intelligent strategies for charging scheduling are essential. Specifically, it is crucial to set an appropriate charging target, which is the target State of Charge (SoC) level at the end of a charging period. A high value may lead to unnecessary charging and high charging cost, while a low value could result in battery depletion during a trip.

The optimization of EB charging schedules has been extensively explored, with existing contributions generally categorized into deterministic and stochastic schemes based on their system modeling approaches. Many studies, such as [3], [4], fall into the deterministic category. They consider system models where the parameter values are deterministic or constant, and typically derive optimal policies through solving Mixed Integer Linear Programming (MILP) problems.

While deterministic models do simplify the problem formulation and solution processes, they struggle to address two significant types of uncertainties encountered in practical scenarios, i.e., uncertainties in EB operation, such as random variations in travel time and energy consumption; plus uncertainties in the smart grid, including fluctuations in electricity prices. Consequently, system models that incorporate these uncertainties and stochastic elements offer a more realistic and applicable representation of the real world, ultimately leading to more reliable and efficient charging schedules.

However, the realistic consideration of uncertainties exacerbates the complexities of finding optimal charging targets. Firstly, since the amount of energy consumption during each trip depends on the time-varying traffic conditions along the bus route, the charging targets must be high enough to support the full trip taking into account these variations. For example, the charging targets have to be higher during rush hours, since it will take the EBs longer time and more energy to complete the trip. Secondly, the charging cost can be reduced by leveraging the real-time electricity prices. For example, EBs can create charging schedules that favor off-peak hours for charging, while discharging energy to the grid during peak hours in the Vehicle-to-Grid (V2G) mode [5].

In recent years, some studies within the stochastic category, such as [6]–[11], have explored the uncertainties in EB travel time and energy consumption, utilizing methodologies such as Robust Optimization (RO) [8]–[11], Immune Algorithms (IA) [6], and Genetic Algorithms (GA) [7]. These approaches aim to enhance adaptability to the variable nature of EB operations. However, existing studies generally do not account for the uncertainties in real-time electricity prices, limiting their ability to optimize charging targets for different charging periods. Additionally, many existing investigations assume a constant charging power throughout the entire charging period, restricting EBs from discharging to the power grid via V2G and adjusting charging power in real time based on fluctuating electricity prices—both of which could further reduce charging costs.

To address the above limitations, this paper develops a charging strategy for EBs based on Deep Reinforcement Learning (DRL). DRL combines Reinforcement Learning (RL) with Deep Neural Networks (DNNs), and has shown significant potential to address uncertainties in operational environments in recent years. Compared with other techniques of dealing with uncertainty, such as RO, IA, and GA, DRL learns optimal policies directly from interactions with the environment, without relying on predefined stochastic models of the variables [12]. Moreover, DRL dynamically learns and updates policies in real time, allowing it to swiftly adapt to changes in dynamic environments. Despite its great potential, there is currently a scarcity of literature applying DRL to charging scheduling for EBs [13]–[17], mainly due to the difficulty in designing an efficient and stable learning process.

Unlike existing DRL-based studies for EB charging, our work accounts for uncertainties in both electricity prices and traffic conditions while also incorporating flexible charging power decisions on a fine time-scale - such as every few minutes - to minimize the charging costs while ensuring sufficient battery energy for EB operations. Unfortunately, this task corresponds to a long-range multi-phase planning problem, which has not been addressed by the aforementioned DRL-based works. The solution to this problem is a key challenge in DRL, where the long action sequences give rise to issues such as slow convergence, high complexity, and insufficiently diverse exploration [18]. The situation further deteriorates when rewards are sparse, as in the EB charging problem when the penalty for running out of battery during trips occurs sporadically. As a result, learning from this undesirable outcome is challenging, because of its rare occurrence.

To overcome the aforementioned challenges, we design a novel Hierarchical DRL (HDRL) algorithm [19] in this paper. The main contributions are summarized as follows.

- 1) **System Model:** Our system model extends beyond the bus transit system to include interactions with the electrical grid, embracing the uncertainties of electricity prices and integrating V2G capabilities. Furthermore, charging scheduling decisions at a finer time-scale are considered, which determine the amount of real-time charging power, rather than relying on fixed charging power levels.
- 2) **HDRL Model:** To address the challenges in solving the long-range multiple-phase planning problem with sparse reward, the charging target option is defined, which simultaneously serves as a temporally extended action and a closed-loop policy. By introducing options, the original MDP is decomposed into a novel two-level decision process including a high-level Semi-MDP (SMDP) and low-level MDPs. We demonstrate that effective policies can be learned from solving the

high-level SMDP for prescribing data-driven charging targets. In order to learn a charging scheduling policy to realize the charging targets whenever possible with minimal charging cost, we define an intrinsic reward for the low-level MDPs. It is proved that the flat policy created by superimposing the optimal high-level policy and the optimal low-level policy performs as well as the optimal policy of the original MDP. Meanwhile, since the long-range effect is captured by the options, the low-level MDPs become isolated from each other, which can be trained over a shorter time horizon to improve the learning efficiency of long-range planning.

- 3) HDRL Solution: A novel algorithm, namely the so-called Hierarchical Double Deep Q Network (HDDQN)-Hindsight Experience Replay (HER) is proposed for simultaneously solving the high-level SMDP and low-level MDPs while capturing the interplay between them. Specifically, both the high-level and low-level agents adopt the DDQN framework [20], but along with separate experience replay buffers. Both agents observe the system states and rewards from the environment for constructing their own rewards. The high-level agent prescribes specific charging target options that are fed to the low-level agent, which decides on the charging powers that are implemented in the environment. Finally, novel approaches based on HER are applied in order to reduce the impact of non-stationarity on the high-level agent and sample inefficiency on the low-level agent brought about by simultaneously learning multiple levels of policies. This innovation enables the proposed algorithm to learn more efficiently and converge faster.

The rest of the paper is organized as follows. Section II critically appraises the state-of-the-art. The system model and MDP model are introduced in Sections III and IV, respectively, while the two-level decision process with options is formulated in Section V. The HDDQN-HER algorithm is proposed in Section VI. Finally, our experiments are highlighted in Section VII, and conclusions are offered in Section VIII.

II. STATE-OF-THE-ART

A. RESEARCH ON EB CHARGING SCHEDULING PROBLEM

There are three major charging technologies for EBs: (i) conductive charging; (ii) battery swapping; and (iii) wireless charging. The studies on conductive charging scheduling can be further divided into plug-in charging and pantograph charging. Both types are classified into two distinct approaches: (i) depot charging which adopts normal or slow chargers to charge EBs at bus depots overnight; (ii) opportunistic charging which utilizes fast chargers to charge

EBs at terminal stations or stops [21]. In the following, we will primarily review the existing literature on the EB Charging Scheduling Problem (EBCSP) for plug-in charging and opportunistic charging, which constitutes the main focus of this paper.

Traditionally, all the parameters in the EBCSP, such as the travel time and energy consumption for each trip, are assumed to be constant or known. Based on this deterministic information, the EBCSP is normally formulated as a MILP problem and solved by different methods such as Branch & Price (BP) [22], the column generation based algorithm [23], dynamic programming [24], and optimization solvers such as CPLEX [3], [4], [25]–[27]. Some studies [3], [26] fully charge EBs before embarking on a new trip, while other studies [22], [23], [28] opted for partial charging, where EBs can charge to varying SoC levels during different charging periods. In realistic scenarios, the travel time and energy consumption are inherently stochastic due to factors such as traffic flow and intersection waiting time. Therefore, their exact values are unknown when solving the EBCSP in practice. Since MILP relies on deterministic modeling, it struggles to effectively handle these uncertainties. To address this issue, He et al. [3] estimated an EB's travel time and energy consumption using the K-means clustering algorithm, and plugged in the predicted values in the MILP model to derive the charging schedule. However, it is impossible to completely eliminate the prediction errors given current technology limitations.

In order to address the uncertainties associated with stochastic travel time and/or energy consumption, stochastic models have been formulated. These models inherently increase the complexity of the optimization algorithms designed for charging scheduling. For example, RO was widely used by the studies [8], [9], [10], and [11], to achieve the dual goals of reducing charging costs and improving system robustness. Although RO can handle uncertainties and provide robust solutions within a predefined uncertainty set, it often leads to overly conservative policies, sacrificing performance to mitigate risks. Some studies opted for heuristic algorithms to address the uncertainties. For example, Liu et al. [6] employed the ant colony algorithm and IA to optimize charging plans under fixed and stochastic travel times, respectively. Bie et al. [7] presented the probability distribution function of trip energy consumption and adopted GA to reduce the charging costs as well as the delay in trip departure times. Compared to MILP, heuristic algorithms represented by GA have lower computational costs. However, they require multiple iterative optimizations, resulting in slow convergence.

Unlike traditional methods, DRL does not require precise modeling; instead, it learns directly from interactions with the environment, making it the most effective in handling uncertainty. In Fig. 1, we visually present the comparison of MILP, RO, GA, and DRL. There is relatively little research on DRL-based solutions for EBCSP. Among these studies,

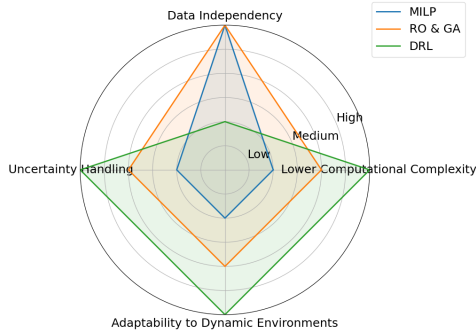


FIGURE 1: Comparison of MILP, RO, GA, and DRL in terms of four key criteria: lower computational complexity, data independency, uncertainty handling, and adaptability to dynamic environments.

Wang et al. [15] combined clipped Double Q-learning (DQL) with Soft Actor-Critic (SAC) to simultaneously solve EB dispatching and charging scheduling problems, where the charging scheduling decision is a simple binary variable. Yan et al. [14] adopted Q-Learning and Twin Delayed Deep Deterministic Policy Gradient (TD-DDPG) to determine the EBs' target SoC levels and their assigned service routes, while assuming fixed charging power. Chen et al. [13] used DQL to make decisions regarding the charging power for EBs upon their arrivals at the terminal station. However, the charging power remains constant for each charging period. By contrast, our work considers more flexible charging power decisions at a smaller time-scale. In summary, while prior works apply various classical DRL techniques, they face challenges due to the inherent long-range multi-phase planning problem when applied to our system model.

Table 1 boldly contrasts our work to the existing literature on EBCSP in terms of key features. While most existing work aims for minimizing the charging cost and considers the more flexible partial charging option rather than full charging, only some of the studies take into account the uncertainties in energy consumption and/or travel time, and consider the charging power as an adjustable decision variable. In addition, existing treatises generally do not incorporate more advanced smart grid characteristics, such as the V2G mode and uncertainty in electricity price. Furthermore, the DRL-based algorithms are rarely adopted to solve the EBCSP, addressing the uncertainties in practical systems. As shown in Table 1, our paper considers all the listed features, filling the corresponding gaps in this field.

B. HIERARCHICAL RL (HRL)

The existing approaches on HRL are mainly developed based on three basic frameworks [31], i.e., the option framework by Sutton et al. [32], MAXQ by Dietterich et al. [33], and

the Hierarchy of Abstract Machines (HAMs) by Parr and Russell [34]. The studies on the option framework can be categorized into the approaches with or without subtask discovery. The approaches with subtask discovery, such as those in [35], [36], do not predefine the subtasks for the options. By contrast, the approaches without subtask discovery, such as the H-DQN by Kulkarni et al. [37], associate each option with a subgoal state for the low-level policy to achieve. In this paper, we adopt the option framework without subtask discovery and associate the options with the charging targets. In contrast to the existing HRL algorithms operating without subtask discovery [37]–[39], we formally formulate our problem associated with the options, providing a unified framework for both option learning and subgoal learning.

An important challenge when simultaneously learning a hierarchy of policies is associated with the non-stationarity [19], since the low-level policy may change during the training process, which leads to non-stationary reward functions and transition probabilities of the high-level SMDP. To address this challenge, the Hierarchical Reinforcement Learning with Off-policy Correction (HIRO) by Nachum et al. [38] adopted a mechanism of subgoal re-labeling. The Hierarchical Actor-Critic (HAC) regime of Levy et al. [39] integrates the ideas of Hindsight Experience Replay (HER) [40] with HIRO for further improvement. Since our objective in this paper involves not only achieving the charging target subgoals but also minimizing the charging cost, our problem is more challenging compared to the studies in [38] and [39]. Fig. 2 shows the contributions of this paper in contrast to the literature of HRL.

III. SYSTEM MODEL

The variables and parameters used in this paper are summarized in Table 2. In general, we use cursive letters to represent a set, such as \mathcal{S} and \mathcal{A} , uppercase letters to represent the total number of variables in a set, such as T and K , and the corresponding lowercase letters to represent the variable indexes in a set, such as t and k . In addition, following the convention of reinforcement learning, we generally use the capital letters with subscript t to denote random variables, such as S_t and A_t , while the corresponding lowercase letters represent sample values of the random variables, such as s and a . We use uppercase letters to represent constant values, such as E_{\max} and E_{\min} . Finally, we use lowercase letters to represent the functions, such as $r(S_t, A_t)$.

In this paper, we consider an EB routinely traveling along a bus route that includes several bus stops and a terminal station. When the EB arrives at the terminal station, its battery pack can be charged. Therefore, the status of the EB can be divided into two alternating periods. One is the charging period, in which the EB stays at the terminal station for charging. The other is the operating period, in which the EB travels along the designated route.

TABLE 1: Contrasting this paper to the literature on EBCSP.

Features	[3], [26]	[22]–[25] [28]–[30]	[27]	[4]	[6]–[9]	[10], [11]	[14], [15]	[13]	Our work
Charging cost minimization	✓	✓	✓	✓	✓	✓			✓
Partial charging		✓	✓	✓	✓	✓	✓	✓	✓
Uncertainties in energy consumption and/or travel time					✓	✓	✓	✓	✓
Adjustable charging power			✓			✓		✓	✓
DRL-based algorithm							✓	✓	✓
V2G mode				✓					✓
Uncertainty in electricity price									✓

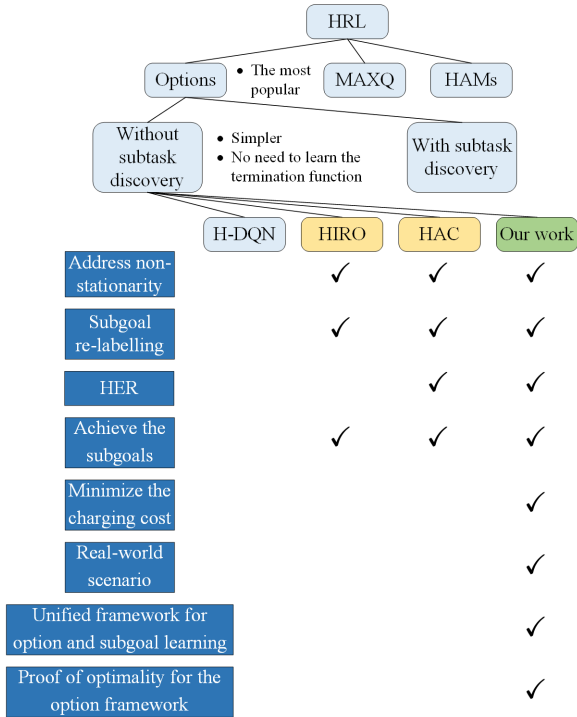


FIGURE 2: The contributions of this paper in contrast to the literature on HRL.

As shown in Fig. 3, consider that there are K operating periods for the EB in a single day according to its bus schedule, where each operating period is indexed by $k \in \{0, \dots, K-1\}$. Additionally, each operating period $k \in \{0, \dots, K-2\}$ is followed by a charging period with the same index k . Given our focus on opportunistic charging, which involves fast chargers at terminal stations, we exclude depot charging that typically occurs overnight after the last operating period $K-1$ of the day. Therefore, the time horizon considered in this paper starts and ends with an operating period, and there are $K-1$ charging periods sandwiched between K operating periods.

The time in a full day is discretized into T equal-length

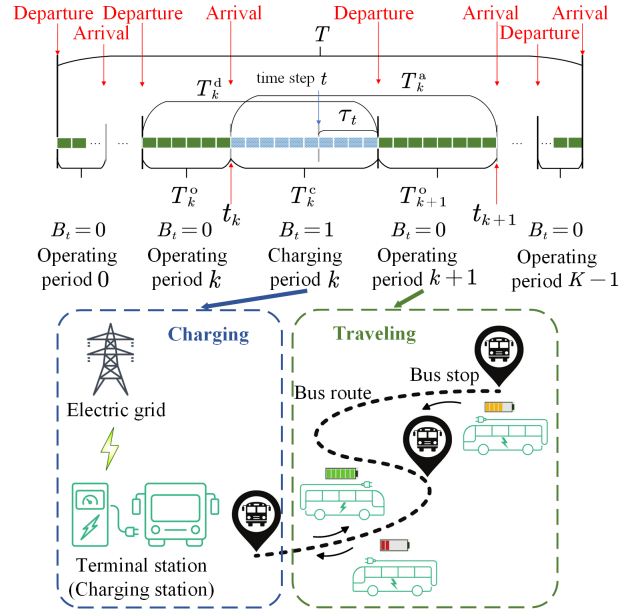


FIGURE 3: The schematic diagram of the system model for the EB charging problem.

time steps, indexed by $\{0, \dots, T-1\}$. The duration of each time step is denoted as Δt . Let $T_k^o, \forall k \in \{0, \dots, K-1\}$ and $T_k^c, \forall k \in \{0, \dots, K-2\}$ be random variables that represent the number of time steps in the operating period k and the charging period k , respectively. Note that the variables T_k^o and T_k^c are random due to the uncertainty in the arrival times of the EBs at the terminal station, which is caused by the stochastic traffic conditions along the bus route. The random time step at which the EB arrives at the terminal station from the operating period k is denoted by t_k , where $k \in \{0, \dots, K-1\}$. Then, the EB enters the charging period k upon its arrival at t_k , where $k \in \{0, \dots, K-2\}$.

Let $T_k^d, \forall k \in \{0, \dots, K-2\}$ denote the fixed number of time steps between two consecutive departures of the EB for operating periods k and $k+1$ according to the bus schedule. This corresponds to the sum duration of a pair of consecutive

TABLE 2: Notation used in this paper

Notations	Description
Sets	
\mathcal{A}	The action space
\mathcal{I}_ω	The initiation set of states for option ω
$\mathcal{S}^+, \mathcal{S}^T / \mathcal{S}$	The state space, the set of terminal/non-terminal states
Ω	The option space
Parameters	
B_t	The EB status at time step t , 0 for operating periods and 1 for charging periods
C^{end}	The penalty for EB to enter the terminal states
C_{\max} / D_{\max}	The maximum absolute value of charging/discharging power of EBs
E_t	The SoC level of the battery at time step t
E_{\max} / E_{\min}	The maximum/minimum storage constraint of the battery
H_t	The historical electricity prices in the period spanning from time step $t - w_p$ up to time step t
K	The number of operating periods in a day
k	The index of an operating/charging period
k_t	The index of the current charging/operating period at the time step t
P_t	The electricity price at time step t
S_t	The system state at time step t
t_k	The random time step at which the EB arrives at the terminal station from the operating period k
T_k^a	The random number of time steps between two consecutive arrivals of the EB from operating periods k and $k + 1$ at t_k and t_{k+1} , respectively
T_k^d	The fixed number of time steps between two consecutive departures of the EB for operating periods k and $k + 1$ according to the bus schedule
T_k^c / T_k^o	The random number of time steps in the charging/operating period k
w_p	The length of the time window to look into the past prices
Δt	The duration of each time step
τ_t	The number of remaining time steps from time step t to the next departure time
Decision Variables	
A_t	The action at time step t
C_t	The charging/discharging power of the battery at time step t
ω_t	The charging target option at time step t
Functions	
$c^{\text{tar}}(S_t, \omega_t)$	The penalty of not realizing the charging target
$r(S_t, A_t)$	The reward function
$r^H(S_t, \omega_t)$	The expected cumulative reward within one transition of SMDP
$r^L(S_t, \omega_t, A_t)$	The intrinsic reward of the low-level MDPs
$\beta_\omega(S_t)$	The termination condition of option ω
$\Gamma_t(B_t)$	The probability that the current period is terminated at time step t
$\pi_\omega(S_t)$	The intra-option policy for option ω
$\mu(S_t)$	The policy over options

operating and charging periods, i.e., $T_k^d = T_k^o + T_k^c$. Let $T_k^a, \forall k \in \{0, \dots, K - 2\}$ represent the random number of time steps between two consecutive arrivals of the EB from operating periods k and $k + 1$ at t_k and t_{k+1} , respectively. This corresponds to the sum duration of a pair of consecutive charging and operating periods, i.e., $T_k^a = t_{k+1} - t_k = T_k^c + T_{k+1}^o$.

We define B_t as a random variable that represents the EB status at time step t , where $B_t = 1$ and $B_t = 0$ correspond to the charging and operating periods, respectively. In addition,

let k_t denote the index of the current charging or operating period at time step t ¹.

Let τ_t denote the number of remaining time steps from time step t to the next departure time. The iterative calculation for τ_{t+1} is derived as

$$\tau_{t+1} = \begin{cases} T_{k_{t+1}}^d, & \text{if } B_{t+1} = 0 \text{ and } B_t = 1 \\ \tau_t - 1, & \text{otherwise} \end{cases}, \quad (1)$$

¹To avoid potential confusion, we would like to highlight that t_k is the index of a time step and k_t is the index of a charging/operating period.

which means that τ_t decreases by 1 at each time step until it reaches 0. When $\tau_t = 0$, the EB will depart for operating period k_{t+1} at the next time step $t + 1$. In this case, τ_{t+1} is set to a fixed value $T_{k_{t+1}}^d$ according to the bus schedule.

We use $\Gamma_t(B_t)$ to denote the conditional probability that the current period is terminated at time step t , given whether the current period is charging (i.e., $B_t = 1$) or operating (i.e., $B_t = 0$). Note that $\Gamma_t(B_t)$ depends on τ_t and k_t . Intuitively, in the operating period ($B_t = 0$), the probability of the EB returning to the terminal station increases as its traveling time elapses. Mathematically, we have

$$\Gamma_t(B_t = 0) = \Pr(B_{t+1} = 1 | B_t = 0, \tau_t, k_t) = \frac{\Pr(T_{k_t}^o = T_{k_t}^d - \tau_t)}{\prod_{x=0}^{T_{k_t}^d - \tau_t - 1} [1 - \Pr(T_{k_t}^o = x)]}, \quad (2)$$

where the numerator represents the probability of the EB arriving at the terminal station at time step t , while the denominator represents the probability of the EB not arriving at the terminal station before time step t .

For the charging period ($B_t = 1$), we consider that the bus line works in accordance with a fixed schedule and the EBs must depart on time. Therefore, when the scheduled departure time is reached, i.e., $\tau_t = 0$, the charging period terminates with $\Gamma_t = 1$. Meanwhile, we have $\Gamma_t = 0$ in all the other time steps. Therefore, we have

$$\Gamma_t(B_t = 1) = \Pr(B_{t+1} = 0 | B_t = 1, \tau_t) = \begin{cases} 1, & \text{if } \tau_t = 0 \\ 0, & \text{otherwise} \end{cases}. \quad (3)$$

The optimization objective is to determine the EB charging schedule in the charging period that minimizes the charging cost while keeping a sufficient SoC level in the battery to ensure reliable operation in the operating period. The charging cost depends on real-time electricity prices, which fluctuate over time. At each time step t , we use P_t to denote the electricity price.

Let E_t denote the SoC level of the battery for the EB at time step t , which is constrained by

$$E_{\min} \leq E_t \leq E_{\max}. \quad (4)$$

Furthermore, let C_t denote the charging/discharging power of the EB battery at time step t , which is positive when charging and negative when discharging. Note that the charging schedule C_t only has to be determined in the charging period, where the EB can either draw energy from or return energy back to the electric grid in the V2G mode. On the other hand, the EB is always discharging in the operating period when the EB travels along the route, and thus C_t is negative. Expressed

mathematically, the constraint of C_t can be written as:

$$C_t \in \begin{cases} [-D_{\max}, C_{\max}] \cap [\frac{E_{\min} - E_t}{\Delta t}, \frac{E_{\max} - E_t}{\Delta t}], & \text{if } B_t = 1 \\ [-D_{\max}, 0] \cap [\frac{E_{\min} - E_t}{\Delta t}, 0], & \text{if } B_t = 0 \end{cases}, \quad (5)$$

where C_{\max} and D_{\max} denote the maximum absolute value of charging and discharging power, respectively. In addition, $[(E_{\min} - E_t) / \Delta t, (E_{\max} - E_t) / \Delta t]$ represents the value range due to the limitation of the EB battery capacity. Finally, the dynamics of the EB battery can be modeled as

$$E_{t+1} = E_t + C_t \Delta t. \quad (6)$$

IV. MDP MODEL

A. STATE

Let $S_t = (E_t, B_t, \tau_t, H_t, k_t)$ be the system state at time step t . H_t denotes the historical electricity prices in the period spanning from time step $(t - w_p)$ up to time step t , i.e.,

$$H_t = (P_{t-w_p}, P_{t-w_p+1}, \dots, P_{t-1}, P_t). \quad (7)$$

Note that w_p is the length of the time window used for considering past prices.

Let S^+ denote the state space, which can be divided into the set of non-terminal states $\mathcal{S} = \{S_t | E_t \geq E_{\min}\}$ and the set of terminal states $\mathcal{S}^T = \{S_t | E_t < E_{\min}\}$. When the SoC level in the EB's battery is lower than the minimum battery capacity constraint, i.e., $E_t < E_{\min}$, the agent will enter the terminal states and the current episode will end before the maximum time step T is reached.

B. ACTION

Let $A_t = C_t \in \mathcal{A}$ be the action at time step t , where \mathcal{A} represents the action space. The charging scheduling policy determines the action A_t only in those states S_t that are associated with $B_t = 1$. The action space \mathcal{A} can be derived based on the first case in (5). When the EB is in the operating period ($B_t = 0$), C_t is a random variable whose value at each time step t is given by the environment rather than being determined by the agent. The range of the random variable C_t is specified by the second case in (5).

C. TRANSITION PROBABILITY

The state transition probability is derived as

$$\Pr(S_{t+1} | S_t, A_t) = \Pr(B_{t+1} | B_t, \tau_t, k_t) \Pr(k_{t+1} | k_t, B_t, B_{t+1}) \Pr(\tau_{t+1} | \tau_t, B_t, B_{t+1}, k_{t+1}) \Pr(E_{t+1} | E_t, A_t) \Pr(H_{t+1} | H_t), \quad (8)$$

where the transition probability of state S_t is decomposed into the product of the transition probabilities of each component in the state definition. The transition probabilities of

historical electricity prices $\Pr(H_{t+1}|H_t)$ are not available, but samples of the trajectory can be obtained from real-world data. The transition probability of SoC level, i.e., $\Pr(E_{t+1}|E_t, A_t)$ can be calculated by (6). Next, the transition probability $\Pr(B_{t+1}|B_t, \tau_t, k_t)$ can be derived from the termination probability $\Gamma_t(B_t)$, i.e.,

$$\Pr(B_{t+1}|B_t, \tau_t, k_t) = \begin{cases} 1 - \Gamma_t(B_t), & \text{if } B_{t+1} = B_t \\ \Gamma_t(B_t), & \text{if } B_{t+1} = 1 - B_t \end{cases}, \quad (9)$$

where $\Gamma_t(B_t)$ is given by (2) and (3). The transition probability of k_t , i.e., $\Pr(k_{t+1}|k_t, B_t, B_{t+1})$, is derived by the iterative calculation for k_{t+1} , i.e.,

$$k_{t+1} = \begin{cases} k_t + 1, & \text{if } B_{t+1} = 0 \text{ and } B_t = 1 \\ k_t, & \text{otherwise} \end{cases}, \quad (10)$$

where $k_0 = 0$. Finally, the iterative calculation for τ_{t+1} in (1) can be used for calculating $\Pr(\tau_{t+1}|\tau_t, B_t, B_{t+1}, k_{t+1})$.

D. REWARD FUNCTION

The optimization objective is to minimize the charging costs, while guaranteeing sufficient energy in the EB battery during the operating period. Therefore, we define the reward function as

$$r(S_t, A_t) = \begin{cases} -C_t \Delta t P_t B_t, & \text{if } S_t \in \mathcal{S} \\ -C^{\text{end}}, & \text{if } S_t \in \mathcal{S}^T \end{cases}, \quad (11)$$

where $C_t \Delta t P_t B_t$ is the charging cost and C^{end} is the penalty to the agent for entering the terminal states. This means that the EB battery is not sufficiently charged to support the entire operating period to completion. As a result, the EBs having depleted batteries receive more negative expected cumulative rewards than their counterpart associated with normal operation, due to receiving a large negative reward $-C^{\text{end}}$ upon entering the terminal state. In practice, C^{end} is a hyperparameter tuned empirically. If set too small, the agent risks reaching the terminal state, whereas a value that is too large may lead to an overly conservative policy, increasing charging costs. In our experiments, the optimal C^{end} is typically around 5 to 10 times $C_t \Delta t P_t B_t$, ensuring that the agent consistently avoids terminal states while still optimizing for lower charging costs.

E. OPTIMIZATION OBJECTIVE

Define the return G_t as the sum of rewards starting from time step t , i.e.,

$$G_t = \sum_{t'=t}^{T-1} r(S_{t'}, A_{t'}). \quad (12)$$

The state-value or value function $V_\pi(s)$ is defined as the expected return from starting in state s and then following

policy π , i.e.,

$$V_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s], \forall s \in \mathcal{S}. \quad (13)$$

Note that the value function of terminal states is always $-C^{\text{end}}$, i.e., $V_\pi(s) = -C^{\text{end}}, \forall s \in \mathcal{S}^T$. Thus, our objective is to derive the optimal policy π^* that maximizes the value function for all the non-terminal states

$$V_{\pi^*}(s) = \max_{\pi} V_\pi(s), \forall s \in \mathcal{S}. \quad (14)$$

In order to derive the optimal policy, the action-value function or Q function $Q_\pi(s, a)$ is defined as the expected return from starting in state s , taking action a , and then following policy π , i.e.,

$$Q_\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a], \forall s \in \mathcal{S}, a \in \mathcal{A}. \quad (15)$$

Note that the Q function for the terminal states is always $-C^{\text{end}}$, i.e., $Q_\pi(s, a) = -C^{\text{end}}, \forall s \in \mathcal{S}^T, a \in \mathcal{A}$. The optimal action-value function is then $Q_\pi^*(s, a) = \max_{\pi} Q_\pi(s, a)$. Given $Q_\pi^*(s, a)$, the optimal policy selects the highest-valued action in each state.

In order to estimate $Q_\pi^*(s, a)$ typically, the following Bellman Equation is used as an iterative update, where

$$Q^*(s, a) = \mathbb{E} \left[r(s, a) + \max_{a'} Q^*(s', a') \right]. \quad (16)$$

The formulated EB charging problem corresponds to an episodic task with a maximum of T time steps in an episode. An episode terminates earlier if the terminal state is reached, which means that the EB runs out of battery. For any full episode where the EB does not run into the terminal state, the last period is always the charging period $K - 1$, where the EB has to charge sufficient energy for the operating period 0 of the next day. The optimal policy is stationary for any episodic task [32], [41].

V. HIERARCHICAL DECOUPLING VIA THE OPTIONS FRAMEWORK

The MDP model defined in Section IV is essentially a long-range multiple-phase planning problem, since the time horizon is divided into multiple charging and operating periods, where each phase consists of a large number of time steps. In the following, we adopt the framework of options [32], [35] to tackle this problem, which enables the agent to abstract actions at different temporal levels.

A. DEFINITION OF OPTIONS

Let $\omega \in \Omega$ denote the options, where Ω is the option space. Options can be regarded as temporally extended “actions”, which can last for multiple time steps. An option is prescribed

by the policy over options μ according to the current state S_t . Each option ω is associated with the triple of $(\mathcal{I}_\omega, \pi_\omega, \beta_\omega)$, where \mathcal{I}_ω is the initiation set of states, i.e., ω is available in state S_t if and only if $S_t \in \mathcal{I}_\omega$, while π_ω is the intra-option policy and β_ω is the termination condition. If an option ω is chosen at time step t , the option will terminate with probability $\beta_\omega(S_{t+l})$ at each time step $t+l$, $\forall l \in 1, 2, \dots$. If the option ω does not terminate at time step $t+l$, $\forall l \in 0, 1, 2, \dots$, an action A_{t+l} is chosen according to the intra-option policy $\pi_\omega(S_{t+l})$ and the environment moves on to the next time step $t+l+1$. Otherwise, a new option ω' may be selected according to the policy over options $\mu(S_{t+l})$, and the same process is repeated for the following time steps.

In this paper, we define the charging target option ω , which corresponds to the target SoC level of the EB's battery, when a charging period terminates. Note that the option space is defined by the E_t constraints in (4). Since the charging targets only have to be determined in the charging periods, \mathcal{I}_ω is defined as

$$\mathcal{I}_\omega = \{S_t \in \mathcal{S} | B_t = 1\}. \quad (17)$$

The termination condition is defined as

$$\beta_\omega(S_t) = \begin{cases} 1, & \text{if } B_t = 0 \text{ and } B_{t+1} = 1 \\ 0, & \text{otherwise} \end{cases}, \quad (18)$$

which means the option should be terminated when an operating period ends and the next charging period will start.

B. DECOUPLING THE ORIGINAL MDP

As shown in Fig.4, by introducing options, our original MDP model is divided into low-level and high-level decision processes. The top panel illustrates the original MDP defined in Section IV, where a flat policy π is used for determining the charging schedule action C_t at each time step t during the charging periods. Meanwhile, C_t is given by the environment according to the amount of power discharged during each time step in the operating periods. Therefore, the original MDP consists of an interleaving of MDPs corresponding to charging periods and Markov Reward Processes (MRPs) representing the operating periods.

Although the original MDP can be solved directly to obtain π , a more efficient solution is to divide it into two levels through options. The middle panel shows the high-level SMDP, where the policy over options μ prescribes the charging target option ω at the selected time steps $\{t_k\}_{k=0}^{K-2}$ when the charging periods start. Assume that option ω is selected according to $\mu(S_{t_k})$ at time step t_k when a charging period starts. When the charging period terminates after a random number of T_k^c time steps, the state S_{t_k} transits to $S_{t_k+T_k^c}$, where $E_{t_k+T_k^c}$ should be as close to the desired charging target ω as possible. When the next charging period starts at time step $t_{k+1} = t_k + T_k^a$, the next option ω' is selected according to $\mu(S_{t_{k+1}})$.

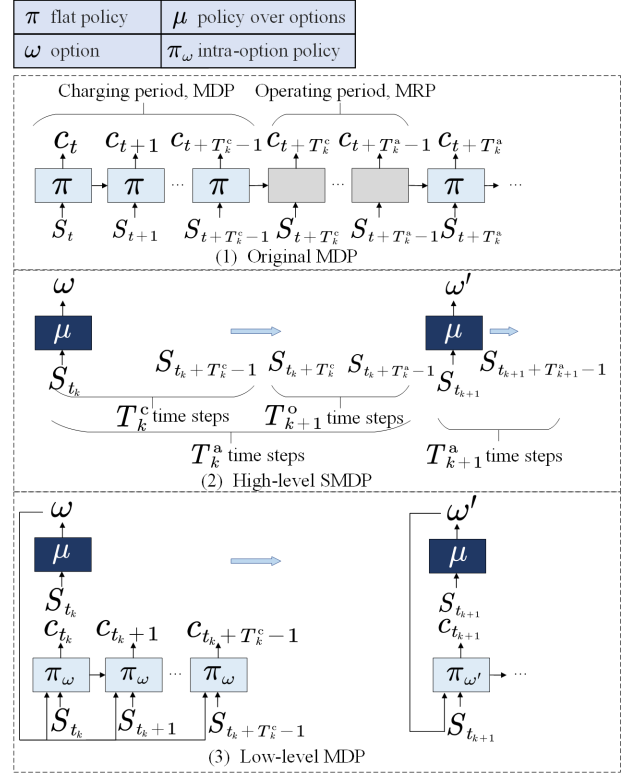


FIGURE 4: The schematic diagram of the original MDP, high-level SMDP, and low-level MDPs.

Finally, the bottom panel shows that the low-level MDPs are embedded into the high-level SMDP, where at each charging period, the intra-option policy π_ω of the selected option ω determines the optimal charging schedule action C_t that minimizes the charging cost, while ensuring that the charging target of ω is realized. By superimposing the high-level policy over options μ and the low-level intra-option policy π_ω , a flat policy is constructed to solve the original MDP problem. Notably, the long time-horizon of the original MDP is divided into multiple shorter time-horizons of the low-level MDPs. Since each low-level MDP is provided with a charging target option obtained by solving the high-level MDP, the low-level MDPs become independent of each other. In other words, each low-level MDP only has to concentrate on its immediate time-horizon for decision-making, without having to look further into the future and consider those low-level MDPs in subsequent time-horizons.

By adopting the hierarchical structure and solving the high-level SMDP and low-level MDPs instead of the original MDP, the learning efficiency and convergence speed can be significantly improved.

C. THE HIGH-LEVEL SMDP

When we only consider the high-level policy over options μ to select options (i.e., assuming the intra-option policy π_ω is available for each option $\omega \in \Omega$), the decision process becomes an SMDP [41] that can be derived from the original MDP.

In SMDP, state transitions and action selections take place at discrete time, but the length of time interval from one transition to the next is random. Therefore, an SMDP consists of state space, action space, an expected cumulative discounted reward for each state-action pair, and a joint distribution of the next state and transit interval. In our high-level SMDP, the states are the same as those of the original MDP, while the actions correspond to the options. For each state transition when option ω is selected, the transit time is T_k^ω time steps. Note that an option terminates when either the EB arrives at the terminal station where $T_k^\omega = T_k^a$, or the EB runs out of battery during the trip where $T_k^\omega < T_k^a$. The SMDP model has its corresponding reward $r^H(s, \omega)$, i.e., the expected cumulative reward within one transition of SMDP in T_k^ω time steps. Let $\mathcal{E}(\omega, s, t_k)$ denote the event that ω starts in state s at time step t_k . The reward $r^H(s, \omega)$ can be derived by

$$r^H(s, \omega) = \mathbb{E}_{\pi_\omega} \left[\sum_{l=0}^{T_k^\omega-1} r(S_{t_k+l}, A_{t_k+l}) | \mathcal{E}(\omega, s, t_k) \right], \quad (19)$$

where the action A_{t_k+l} is derived based on the intra-option policy π_ω for the first T_k^c time steps during the charging period, and then provided by the environment for the following $T_k^\omega - T_k^c$ time steps during the operating period.

The transition distributions of the SMDP $P_{ss'}^\omega$ can be formulated as

$$P_{ss'}^\omega = \sum_{T_k^c=1}^{\infty} P(s', T_k^\omega), \quad (20)$$

where $P(s', T_k^\omega)$ represents the probability that the option terminates after T_k^ω time steps when the state s changes to s' .

Let us define the high-level return G_k^H as the sum of the high-level rewards starting from time step t_k , i.e.,

$$G_k^H = \sum_{k'=k}^{K-1} r^H(S_{t_{k'}}, \omega_{t_{k'}}), \forall k \in \{0, 1, \dots, K-2\} \quad (21)$$

The high-level value function $V_\mu^H(s)$ is defined as the expected high-level return from starting in state s and then following the policy over options μ , i.e.,

$$V_\mu^H(s) = \mathbb{E}_\mu \mathbb{E}_{\pi_\omega} [G_k^H | S_{t_k} = s], \forall s \in \mathcal{S}. \quad (22)$$

Note that the high-level value function of the terminal states

is always $-C^{end}$, i.e., $V_\mu^H(s) = -C^{end}, \forall s \in \mathcal{S}^T$. The objective of the high-level SMDP is to derive the optimal policy over option μ^* that maximizes the high-level value function for all the nonterminal states

$$V_{*}^H(s) = V_{\mu^*}^H(s) = \max_{\mu} V_{\mu}^H(s), \forall s \in \mathcal{S}. \quad (23)$$

In order to derive the optimal policy over options, the option-value function or high-level Q function $Q_\mu^H(s, \omega)$ is defined as the expected return from starting in state s , taking option ω , and then following policy μ , i.e.,

$$Q_\mu^H(s, \omega) = \mathbb{E}_\mu \mathbb{E}_{\pi_\omega} [G_t^H | S_{t_k} = s, \omega_{t_k} = \omega], \forall s \in \mathcal{S}, \omega \in \Omega. \quad (24)$$

Note that the option-value function for the terminal states is always $-C^{end}$, i.e., $Q_\mu^H(s, \omega) = -C^{end}, \forall s \in \mathcal{S}^T, \omega \in \Omega$. Given the optimal option-value function $Q_*^H(s, \omega) = \max_{\mu} Q_\mu^H(s, \omega)$, the optimal policy over options selects the highest-valued option in state s at time step t_k . The Bellman Optimality Equation is

$$Q_*^H(s, \omega) = \mathbb{E} \left[r^H(s, \omega) + \max_{\omega'} Q_*^H(s', \omega') | \mathcal{E}(\omega, s) \right], \quad (25)$$

where $\mathcal{E}(\omega, s)$ denotes option ω being initiated in state s .

D. THE LOW-LEVEL MDP

As illustrated in Fig. 4, the course of the original MDP is divided into an interleaving of low-level MDPs and MRPs, where the time horizon T_k^c (resp. T_k^o) of each low-level MDP (resp. MRP) corresponds to the charging (resp. operating) period in one transition of the high-level SMDP. In order to derive the intra-option policy π_ω , the low-level MDP is formulated below.

In the low-level MDPs, the state is denoted as (S_t, ω_t) , where the state in the original MDP, i.e., S_t , is augmented by including the current option ω_t . Let \mathcal{S}_L^+ denote the low-level state space, which can be divided into the set of low-level non-terminal states $\mathcal{S}_L = \{S_t \in \mathcal{S}, \omega_t \in \Omega | B_t = 1\}$ and the set of low-level terminal states $\mathcal{S}_L^T = \{S_t \in \mathcal{S}, \omega_t \in \Omega | B_t = 0\}$. Note that the low-level MDP terminates whenever the EB enters the operating period, i.e., $B_t = 0$. The actions and the transition probabilities are the same as those of the original MDP, since ω_t remains the same throughout the time horizon of the low-level MDP. Since we should ensure that the charging target of the selected option ω is realized at the end of the current charging period, the intrinsic reward of the low-level MDPs is defined as

$$r^L(s, \omega, a) = \begin{cases} -C_t \Delta t P_t B_t, & \text{if } (s, \omega) \in \mathcal{S}_L \\ -\kappa C^{\text{tar}}(s, \omega), & \text{if } (s, \omega) \in \mathcal{S}_L^T \end{cases}, \quad (26)$$

where

$$c^{\text{tar}}(s, \omega) = (\omega - E_{t_k + T_k^c})^2. \quad (27)$$

Note that $s = S_t$ and $a = \pi_\omega(s)$ when $(s, \omega) \in \mathcal{S}_L$. Although no action is selected when $(s, \omega) \in \mathcal{S}_L^T$, we keep the action in the reward function of the terminal state for notational simplicity. Furthermore, $c^{\text{tar}}(s, \omega)$ in (27) represents the penalty of not realizing the charging target, which is the squared error between the charging target ω and the actual SoC level $E_{t_k + T_k^c}$ at the end of the charging period. Finally, κ in (26) is the coefficient representing the relative importance of minimizing the charging cost versus the penalty $c^{\text{tar}}(s, \omega)$. κ is set to a high enough value to ensure that the charging target is met whenever possible.

Let us define the low-level return G_t^L as the sum of the intrinsic rewards from time step t to the end of the time horizon of the low-level MDP, i.e.,

$$G_t^L = \left[\sum_{t'=t}^{t_k + T_k^c} r^L(S_{t'}, \omega, A_{t'}) \right], \forall t \in [t_k, t_k + T_k^c]. \quad (28)$$

The low-level value function $V_{\pi_\omega}^L(s, \omega)$ is defined as the expected return from starting in state (s, ω) and then following the intra-option policy π_ω , i.e.,

$$V_{\pi_\omega}^L(s, \omega) = \mathbb{E}_{\pi_\omega} [G_t^L | S_t = s, \omega], \forall (s, \omega) \in \mathcal{S}_L. \quad (29)$$

Note that the value function of low-level terminal states is always $-\kappa c^{\text{tar}}(s, \omega)$, i.e., $V_{\pi_\omega}^L(s, \omega) = -\kappa c^{\text{tar}}(s, \omega), \forall (s, \omega) \in \mathcal{S}_L^T$. The objective of the low-level MDP is to derive the optimal intra-option policy π_ω^* that maximizes the low-level value function for all the states

$$V_*^L(s, \omega) = V_{\pi_*}^L(s, \omega) = \max_{\pi_\omega} V_{\pi_\omega}^L(s, \omega), \forall (s, \omega) \in \mathcal{S}_L. \quad (30)$$

In order to derive the optimal intra-option policy, the low-level action-value function or Q function $Q_{\pi_\omega}^L(s, \omega, a)$ is defined as the expected low-level return from starting in state (s, ω) , taking action a , and then following policy π_ω , i.e.,

$$Q_{\pi_\omega}^L(s, \omega, a) = \mathbb{E}_{\pi_\omega} [G_t^L | S_t = s, A_t = a, \omega], \quad (31)$$

$$\forall (s, \omega) \in \mathcal{S}_L, a \in \mathcal{A}.$$

Note that the action-value function for the low-level terminal states is always $-\kappa c^{\text{tar}}(s, \omega)$, i.e., $Q_{\pi_\omega}^L(s, \omega, a) = -\kappa c^{\text{tar}}(s, \omega), \forall (s, \omega) \in \mathcal{S}_L^T, a \in \mathcal{A}$. Given the optimal action-value function $Q_*^L(s, \omega, a) = \max_{\pi_\omega} Q_{\pi_\omega}^L(s, \omega, a)$, the optimal intra-option policy selects the highest-valued action in state (s, ω) . The Bellman Optimality Equation is given as

$$Q_*^L(s, \omega, a) = \mathbb{E} \left[r^L(s, \omega, a) + \max_{a'} Q_*^L(s', \omega, a') \right]. \quad (32)$$

Theorem 1. *The flat policy created by superimposing the optimal high-level policy over options μ^* and the optimal*

low-level intra-option policy π_ω^ performs as well as the optimal policy of the original MDP π^* , i.e.,*

$$V_{\pi^*}(s) = \mathbb{E}_{\mu^*} \mathbb{E}_{\pi_{\omega^*}^*} \left[\sum_{t=0}^{T-1} r(S_t, A_t) | S_0 = s \right], \forall s \in \mathcal{S}, \quad (33)$$

where $\omega^* = \mu^*(s), \forall s \in \mathcal{S}$ represents the charging targets prescribed by μ^* . Moreover, μ^* is the optimal high-level policy over options when the optimal low-level intra-option policy π_ω^* is available in the high-level SMDP.

VI. DRL SOLUTION

In this section, we present a DRL algorithm, namely the HDDQN-HER conceived for learning the optimal policy over options μ^* by solving the high-level SMDP; and the optimal intra-option policy π_ω^* by solving the low-level MDPs.

A. OVERALL TRAINING FRAMEWORK

We adopt two deep Q networks, i.e., $\mathcal{Q}^H(\cdot; \theta^H)$ and $\mathcal{Q}^L(\cdot; \theta^L)$ with parameters θ^H and θ^L to approximate $Q_*^H(s, \omega)$ and $Q_*^L(s, \omega, a)$, respectively. The two networks are jointly trained, where the overall framework is illustrated in Fig. 5. At each time step t , both the high-level and low-level agents receive state S_t from the environment. The high-level agent only selects a charging target option ω_t based on S_t when a new charging period starts, i.e., $B_t = 1, B_{t-1} = 0$. The option is fed to the low-level agent instead of being fed to the environment. The low-level agent selects a charging schedule action A_t at every time step t in the charging period ($B_t = 1$) based on the state S_t and the option ω_t . The action A_t is fed to the environment, which responds to the agent by providing the reward $r(S_t, A_t)$. Meanwhile, the environment directly provides the agent with the reward at each time step t in the operating period ($B_t = 0$) without requiring an action from the agent. The high-level agent calculates the high-level reward $r^H(S_t, \omega_t)$ by summing up the reward $r(S_t, A_t)$ during each SMDP transition. Meanwhile, the low-level agent derives the low-level reward $r^L(S_t, \omega_t, A_t)$ at each time step based on $r(S_t, A_t)$ and ω_t according to (26) and (27).

The two Q networks $\mathcal{Q}^H(\cdot; \theta^H)$ and $\mathcal{Q}^L(\cdot; \theta^L)$ are trained simultaneously based on the DDQN algorithm. We refer interested readers to [20] for details of the DDQN algorithm. The high-level and low-level agents draw experiences from their own replay buffers, i.e., \mathcal{R}^H and \mathcal{R}^L , respectively. The two replay buffers are updated at different time scales. More specifically, transitions for the low-level MDPs are collected and stored into \mathcal{R}^L at each time step in the time horizon of the low-level MDP. However, only when the agent enters the terminal state or when an operating period terminates (i.e.,

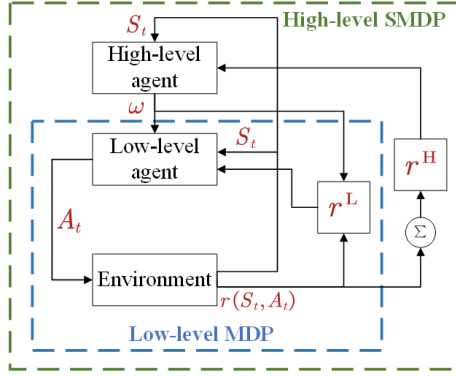


FIGURE 5: The framework of HDDQN-HER algorithm.

$B_t = 0$ and $B_{t+1} = 1$), will a new transition for the high-level SMDP be collected and stored into \mathcal{R}^H .

B. OVERCOMING NON-STATIONARITY IN TRAINING HIGH-LEVEL AGENT

Based on the above framework, it is challenging to directly learn the high-level policy over options μ^* and low-level intra-option policy π_ω^* in parallel. This is mainly due to the non-stationary environment of the high-level agent, which is caused by exploring and updating low-level policy during training. Specifically, the non-stationarity has two facets.

- **Non-stationary reward function:** according to the low-level reward function defined in (26), the low-level agent learns to minimize the cumulative charging cost during the charging period, while ensuring that the charging target is realized. Therefore, the cumulative charging cost decreases, while the low-level policy improves during training, which means that the high-level reward function $r^H(s, \omega)$ as defined in (19) changes over time.
- **Non-stationary transition probabilities:** a sub-optimal low-level policy activated during training might not be able to achieve the charging target ω set by the high-level agent, while the charging target is actually achievable by the optimal low-level policy. In this case, the transition probability $P_{ss'}^\omega$ of the high-level SMDP will change over time as the low-level policy learns to achieve the charging target.

To overcome the above challenges, we adopt the following pair of measures so that the high-level SMDP model remains independent of the changing and exploring low-level policy.

- To deal with the non-stationary reward function, we divide the joint learning process into two phases. In the first phase when the number of training episodes is smaller than a threshold, i.e., $e < M_e$, the high-level rewards \hat{r}^H stored in the high-level replay buffer \mathcal{R}^H are obtained by running a fixed low-level policy π_q instead of those actually achieved by the low-level behavior

policy that is being trained. In the second phase when $e \geq M_e$, we calculate the high-level rewards \hat{r}^H based on the low-level behavior policy, i.e., $\hat{r}^H = r^H$, since the low-level policy almost converges to the optimal one at this phase and the non-stationary reward issue is negligible.

- To deal with the non-stationary transition probabilities, the actual SoC level of the battery $E_{t_k+T_k^c}$ at the end of the charging period is stored in the high-level replay buffer \mathcal{R}^H as the option, instead of the actual option ω that has been prescribed by the high-level agent at time step t_k , when the charging period started. Therefore, the transition stored in \mathcal{R}^H is $[S_{t_k}, \tilde{\omega}, \hat{r}^H(S_{t_k}, \tilde{\omega}), S_{t_k+T_k^a}]$ instead of $[S_{t_k}, \omega, \hat{r}^H(S_{t_k}, \omega), S_{t_k+T_k^a}]$, where $\tilde{\omega} = E_{t_k+T_k^c}$. Note that the next state $S_{t_k+T_k^a}$ is the one that is actually visited in this episode, and the reward $\hat{r}^H(S_{t_k}, \tilde{\omega})$ is calculated as described above, depending on the current training phase.

C. IMPROVING SAMPLE EFFICIENCY IN TRAINING THE LOW-LEVEL AGENT

The low-level agent has to learn to achieve multiple charging targets, while only at the end of an episode will the penalty $c^{\text{tar}}(s, \omega)$ of not realizing a given charging target be available in the low-level reward defined in (26). Both the multi-goal and sparse reward issues impair the sample efficiency in training the low-level agent, which leads to slow convergence in learning the low-level policy, as well as to an adverse effect in overcoming the non-stationarity for the high-level agent. To deal with the above challenges, we harness the following measures for improving the sample efficiency.

- **Restricting the Option Space:** In each charging period, the low-level agent tries to achieve the charging target prescribed by the high-level agent. However, some charging targets might be unachievable, when the high-level agent has not yet learned a good policy. As a result, learning to achieve such charging targets is futile for the low-level agent, and the corresponding experiences are wasted.
- Given the SoC level E_t at the beginning of a charging period, some charging targets are unachievable by the end of the charging period even if the EB is fully charged or discharged at all the time steps. According to the dynamics of the SoC level in (6), a state-dependent option space Ω' can be defined as

$$\Omega' = \{\omega \in \Omega | \omega \in [E_t - \tau_t D_{\max}, E_t + \tau_t C_{\max}]\}. \quad (34)$$

- If a charging target is too low, the EB will run out of battery power during operation and the MDP enters the terminal states. To exclude these unreasonable charging targets, all transitions $(\{-, \omega\}, -, -, \{-, \omega\})$ associated with the current

option ω are deleted from \mathcal{R}^L , whenever the agent enters the terminal states.

- **HER:** In addition to the original transition $(\{s, \omega\}, a, r^L, \{s', \omega\})$ that is stored in the low-level replay buffer \mathcal{R}^L , we create another hindsight transition, i.e., $(\{s, TBD\}, a, \hat{r}^L, \{s', TBD\})$. At the end of the charging period, the actual achieved SoC levels $E_{t_k+T_k^c}$ are filled in the TBD components and the additional hindsight transitions are stored into \mathcal{R}^L if the charging target ω is not realized. Note that $\hat{r}^L = r^L$ at all the time steps, except for the last time step in the time horizon of the low-level MDP, i.e., $B_t = 0$ and $B_{t-1} = 1$, where $c^{\text{tar}}(s, \omega) = 0$ in $\hat{r}^L(s, \omega, a)$ since the substituted charging target $E_{t_k+T_k^c}$ is realized. This method guarantees that at least one sequence of transitions is stored in \mathcal{R}^L that succeeds in realizing the charging target for each charging period.

Algorithm 1 describes the pseudocode of the HDDQN-HER algorithm. The functions to select actions or options, store hindsight transitions into \mathcal{R}^L , and to update the associated Q networks are presented by Algorithms 2, 3, and 4, respectively. In Algorithm 2, the options or actions are prescribed based on the popular ϵ -greedy policy, while in Algorithm 4, the two Q networks $\mathcal{Q}^H(\cdot; \theta^H)$ and $\mathcal{Q}^L(\cdot; \theta^L)$ are updated based on the corresponding sampled minibatch of transitions.

VII. NUMERICAL ANALYSIS

In order to evaluate the effectiveness of the proposed HDDQN-HER algorithm, we perform experiments based on real-world data. All the experiments are performed on a Linux server, where the DRL algorithms are implemented in Tensorflow 1.14 using Python 3.6.

A. EXPERIMENTAL SETUP

1) Simulated Environment

The environment that is simulated in our experiments comprises three EBs serving the same bus route. Each of the EBs operates according to a fixed daily schedule [42]. Specifically, the first EB departs from the terminal station at 6:30 in the morning, and the last one departs at midnight, i.e., 0:00. An EB is scheduled to depart every 30 minutes during this period, which means that the duration between two consecutive departures of each EB is 90 minutes. The duration of the operating period T_k^o follows normal distributions with different mean values at different times of the day, i.e., $T_k^o \sim \mathcal{N}(50, 8)$ at rush hours (7:00-9:00 and 17:00-19:00) and $T_k^o \sim \mathcal{N}(40, 8)$ at other times.

The time horizon for each EB always starts with a charging period, when it arrives at the terminal station after its

Algorithm 1 HDDQN-HER algorithm

```

1: Initialize  $\mathcal{Q}^H(s, \omega; \theta^H)$  and  $\mathcal{Q}^L(s, \omega, a; \theta^L)$  with  $\theta^H = \theta_0^H$  and  $\theta^L = \theta_0^L$ , respectively. Initialize target Q networks  $\mathcal{Q}^{H-}(s, a; \theta^{H-})$  and  $\mathcal{Q}^{L-}(s, \omega, a; \theta^{L-})$  with  $\theta^{H-} \leftarrow \theta^H$  and  $\theta^{L-} \leftarrow \theta^L$ . Initialize  $\mathcal{R}^H$  and  $\mathcal{R}^L$ , exploration probability  $\epsilon^L = 1$  and  $\epsilon^H = 1$ .
2: for episode  $e = 1, \dots, M$  do
3:   Receive the start state  $s$ 
4:   Derive the option space  $\Omega'$  based on (34)
5:    $r^H(s, \omega) \leftarrow 0, s_0 \leftarrow s$ 
6:    $\omega \leftarrow \epsilon$ -GREEDY( $s_0, \Omega', \epsilon^H, \mathcal{Q}^H(s_0, \omega; \theta^H)$ ),  $\tilde{\omega} \leftarrow \omega$ 
7:   for  $t = 1, \dots, T$  do
8:     if  $B_t = 1$  then
9:        $a \leftarrow \epsilon$ -GREEDY( $\{s, \omega\}, \mathcal{A}, \epsilon^L, \mathcal{Q}^L(s, \omega, a; \theta^L)$ )
10:      Execute action  $a$  and observe reward  $r(s, a)$ , and next state  $s'$  from environment
11:      Derive low-level reward  $r^L(s, \omega, a)$ ,
12:      Store  $(\{s, \omega\}, a, r^L, \{s', \omega\})$  into  $\mathcal{R}^L$ 
13:      HER( $s, \omega, a, r^L, s', \tilde{\omega}, \mathcal{R}^L$ )
14:     else
15:       Observe action  $a$ , reward  $r(s, a)$ , and next state  $s'$  from environment
16:     end if
17:     UPDATE( $\mathcal{R}^L, \mathcal{Q}^L(\cdot; \theta^L), \mathcal{Q}^{L-}(\cdot; \theta^{L-})$ )
18:      $r^H \leftarrow r^H + r(s, a)$ 
19:     if  $s$  is terminal state or  $B_t = 0, B_{t+1} = 1$  then
20:       if  $e < M_\epsilon$  then
21:         Derive  $\hat{r}^H$  by running policy  $\pi^q$ 
22:       else
23:          $\hat{r}^H = r^H$ 
24:       end if
25:       Store transition  $(s_0, \tilde{\omega}, \hat{r}^H, s')$  into  $\mathcal{R}^H$ 
26:       UPDATE( $\mathcal{R}^H, \mathcal{Q}^H(\cdot; \theta^H), \mathcal{Q}^{H-}(\cdot; \theta^{H-})$ )
27:       if  $s$  is terminal state then
28:         Delete  $(\{-, \omega\}, -, -, \{-, \omega\})$  from  $\mathcal{R}^L$ 
29:         Terminate this episode
30:       else
31:         Update  $\Omega'$  based on (34)
32:          $\omega \leftarrow \epsilon$ -GREEDY( $s', \Omega', \epsilon^H, \mathcal{Q}^H(s', \omega; \theta^H)$ )
33:          $r^H(s, \omega) \leftarrow 0, s_0 \leftarrow s', \tilde{\omega} \leftarrow \omega$ 
34:       end if
35:     end if
36:      $s \leftarrow s'$ 
37:   end for
38:   Anneal  $\epsilon^L$  and  $\epsilon^H$ 
39: end for

```

first departure in the morning, where a pair of consecutive charging and operating periods corresponds to one transition in the high-level SMDP. The duration of a time step is set to 10 minutes, i.e., $\Delta t = 10$ min. The parameters of the environment are listed in Table 3.

Algorithm 2 Select action a or options ω

```

1: function  $\epsilon$ -GREEDY( $x, \mathcal{B}, \epsilon, \mathcal{Q}(\cdot; \theta)$ )
2:   if random() $< \epsilon$  then
3:     return random element from set  $\mathcal{B}$ 
4:   else
5:     return  $\arg \max_{b \in \mathcal{B}} \mathcal{Q}(x, b; \theta)$ 
6:   end if
7: end function

```

Algorithm 3 Store hindsight transitions

```

1: function HER( $s, \omega, a, r^L, s', \tilde{\omega}, \mathcal{R}^L$ )
2:   Creat hindsight transition
   ( $\{s, TBD\}, a, \hat{r}^L, \{s', TBD\}$ ) where  $\hat{r}^L = r^L$ 
3:   if  $B_{t+1} = 0$  and  $\omega \neq E_{t+1}$  then
4:      $\tilde{\omega} \leftarrow E_{t+1}$ , update reward  $\hat{r}^L$ 
5:     Replace  $TBD$  with  $\tilde{\omega}$  and store hindsight transi-
     tions into  $\mathcal{R}^L$ 
6:   end if
7: end function

```

2) Experimental data

We use real-world data on electricity prices to train the DRL agents. The dataset contains hourly time-varying electricity prices that reflect the wholesale market price derived from the Midcontinent Independent System Operator (MISO) delivery point [43]. Since the electricity price data from one month closely approximates the statistical distribution observed over an entire year, we adopt the one-month electricity price data for training. Specifically, the data in January 2023 is used for training, and the data in the first week of February 2023 is used for testing. Fig. 6 shows the trajectories of electricity prices on a typical day. By observing the fluctuations over time, we find that electricity prices are highest with the value of \$0.03921/kWh during the hours of peak demand (between 17:00 and 18:00 daily).

3) Baseline algorithms

Several baseline algorithms are simulated for performance comparison with HDDQN-HER.

- 1) DDQN-Original: The original MDP of Section III is directly solved by DDQN to obtain a flat policy π .
- 2) DDQN-Low: The low-level MDPs corresponding to the charging periods are solved by DDQN. However, the charging target is set to the maximum battery capacity, as in [3], [26], rather than being determined by the policy over options. The reward function of the low-level MDPs is similar to that defined in (26), except that $\kappa c^{\text{tar}}(s, \omega)$ is replaced by $\kappa' (E_{\max} - E_{t_k + T_k^c})^2$, which measures the range anxiety.
- 3) DDQN-High: The high-level SMDP is solved by

Algorithm 4 Networks updating

```

1: function UPDATE( $\mathcal{R}, \mathcal{Q}(\cdot; \theta), \mathcal{Q}^-(\cdot; \theta^-)$ )
2:   Sample a random minibatch  $\mathcal{M} = \{s^{(i)}, a^{(i)}, r^{(i)}, s'^{(i)}\}$  from  $\mathcal{R}$  and set the target
    $y^{(i)} = r^{(i)} + \mathcal{Q}^-\left(s'^{(i)}, \arg \max_a \mathcal{Q}(s'^{(i)}, a; \theta); \theta^-\right)$ 
3:   Update  $\mathcal{Q}$  by gradient descent on loss  $L(\theta)$ , where
   
$$L(\theta) = \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \left( y^{(i)} - \mathcal{Q}(s^{(i)}, a^{(i)}; \theta) \right)^2,$$

4: end function

```

TABLE 3: Parameter configuration in the system model.

Notations	Values	Description
w_p	4	The length of the time window to look into the past prices
C_{\max} / D_{\max}	120kW / 120kW	The maximum absolute value of charging/discharging power of EBs
E_{\min} / E_{\max}	0kWh / 240kWh	The minimum / maximum storage constraint of the battery
C^{end}	50	The penalty for EB to enter the terminal states
κ / κ'	0.005 / 0.0006	The coefficients of the penalty $c^{\text{tar}}(S_t, \omega_t)$ / the cost of range anxiety

DDQN to learn the charging target. However, the low-level MDPs are solved by a fixed policy π^q rather than by the intra-option policy π_{ω}^* learned by training the low-level agent. Specifically, the charging target is realized whenever possible without considering the fluctuating electricity prices. This approach adapts the charging power at a coarse time scale, specifically once per charging period, similar to the method in [13].

- 4) HDDQN: Both the high-level SMDP and low-level MDPs are solved by two DDQN networks, respectively. This approach adopts the same framework as in Fig. 5, namely the HDDQN-HER. However, the methods of overcoming non-stationarity and improving sample efficiency in Section VI.B and C are not adopted.

Note that a fixed low-level policy π^q is used in DDQN-High as well as the first learning phase in HDDQN-HER. In the experiments, we set π^q as follows: at each time step in a charging period, the charging power c_t is set to the maximum value until the charging target is realized.

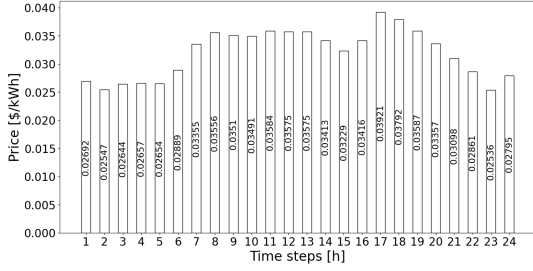


FIGURE 6: The trajectories of the electricity prices of a typical day in the experimental data.

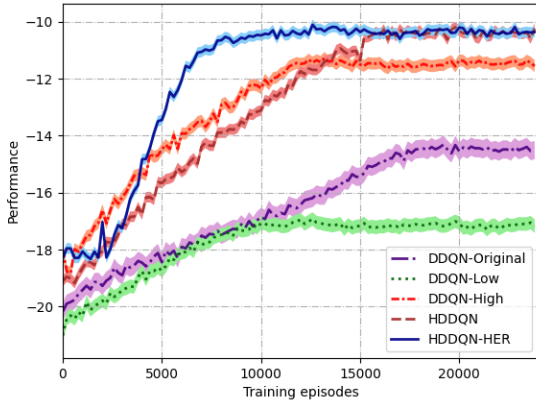


FIGURE 7: The performance curves of the algorithms in the training process. The vertical axis corresponds to the average performance across the three EBs, and the shaded areas represent the standard errors.

The hyper-parameters of the DDQN networks are listed in Table 4. Note that the total number of training episodes is 24000. The training process of HDDQN-HER is divided into two phases with the threshold of $M_e = 6000$. Therefore, the first phase corresponds to when the training episodes obey $e < 6000$, while the second phase corresponds to $6000 \leq e \leq 24000$. The entire training process took approximately 10 hours, while executing a single episode during the deployment process took less than 15 seconds.

B. EXPERIMENTAL RESULTS

1) Performance for the test set

For each EB, we executed 100 complete episodes based on the test set after training. We obtain the individual performance of each EB by averaging its return over the test episodes, where the return of one episode is defined in (12). Table 5 summarizes the individual performance of each EB, as well as the average and maximum performance over the three EBs for all the algorithms. The performance rank-

TABLE 4: Hyper-parameters of various DRL algorithms.

Algorithms	Parameters			
DDQN-	Q network size	Learning Rate	Batch Size	Training Episodes
high-level	256,300,100	5e-6	128	6000+18000
low-level	256,300,100	5e-6	64	6000+18000
Original	256,300,100	5e-6	128	24000
High	256,300,100	5e-6	128	24000
Low	400,300,100	1e-6	64	24000

ings are consistent, where HDDQN-HER achieves the best performance, followed by HDDQN, DDQN-High, DDQN-Original, and DDQN-Low. The average performance of HDDQN-HER is better than those of the baseline algorithms by 0.097%, 9.85%, 28.44%, and 39.60%, respectively.

Both HDDQN and HDDQN-HER achieve much better performance than the other algorithms, demonstrating the effectiveness of our proposed framework in Fig. 5. Note that HDDQN-HER only achieves slightly better average performance than HDDQN, while the maximum performances of the two algorithms are nearly the same. These results show that the proposed methods of overcoming non-stationarity and improving sample efficiency do not improve the final performance of HDDQN-HER much over HDDQN as long as sufficient training episodes are run. However, a significant convergence speed of the training process is observed, as it will be discussed in Section VII.B 2).

DDQN-Original performs worse than both HDDQN-HER and HDDQN, since it struggles to learn an optimal policy by directly solving the original MDP for a long horizon. As the penalty C^{end} for entering the terminal states is sparse, a long sequence of highly-specific actions must be executed prior to observing a terminal reward, which makes the learning task challenging. Meanwhile, the hierarchical framework in HDDQN-HER provides an intrinsic reward for the low-level MDPs through options, which substantially improves the sample efficiency.

HDDQN-HER outperforms DDQN-Low due to the more reasonable charging target in the reward function. The charging target of DDQN-Low is set to the maximum battery capacity, while those of HDDQN-HER are learned by solving the high-level SMDP. Therefore, HDDQN-HER can adapt to the electricity price changes throughout the day and consider the long-range effects of decisions. In addition, compared to HDDQN-HER, DDQN-High discards the low-level DDQN network, which prevents it from adapting EB charging power at a fine time scale to fluctuating electricity prices. Consequently, it cannot learn optimal policies for the low-level

TABLE 5: The individual, average, and maximum performance of the algorithms across three EBs.

Algorithms	Performance				
	EB A	EB B	EB C	Max	Average
HDDQN-HER	-10.23	-10.48	-10.31	-10.23	-10.34
HDDQN	-10.23	-10.50	-10.32	-10.23	-10.35
DDQN-High	-11.25	-11.67	-11.48	-11.25	-11.47
DDQN-Original	-14.08	-14.67	-14.59	-14.08	-14.45
DDQN-Low	-16.92	-17.38	-17.05	-16.92	-17.12

MDPs. Finally, comparisons with both DDQN-High and DDQN-Low demonstrate the importance of both levels of DDQN networks for the proposed algorithm. Omitting either level would hinder the algorithm's ability to achieve optimal performance.

2) Convergence Properties

Fig. 7 shows the performance curves of the algorithms, which are obtained by periodically evaluating the policies during training. Specifically, we ran 10 test episodes for every 100 training episode, where the X-axis is the number of training episodes and the Y-axis is the performance corresponding to the average return over the latest 10 test episodes. The shaded areas indicate the standard errors across the three EBs.

The proposed HDDQN-HER converges first at approximately 8500 episodes, followed by DDQN-Low at approximately 10000 episodes due to its simple framework and short time horizon. However, the performance of DDQN-Low after convergence is inferior to those of the other algorithms. DDQN-High also converges relatively fast at approximately 12000 episodes. By contrast, without the methods of overcoming non-stationarity and improving sample efficiency in HDDQN-HER, HDDQN converges slowly at around 15000 episodes. Finally, the convergence of DDQN-Original is the slowest, which converges at around 18000 episodes. This result further demonstrates the low sample and learning efficiency, when directly solving a long-range planning problem by RL. The shaded areas of all the algorithms are relatively small, which shows that all the algorithms perform consistently for different EBs.

3) Charging Schedule Results

Fig. 8 illustrates the detailed charging schedule decisions and corresponding costs for EB A at each time step in a typical episode based on the test set. The grey and white regions indicate the operating and charging periods, respectively. Fig. 8(a), 8(c), and 8(e) show the charging power at each time step and the EB's battery trajectory. Notice that there is a horizontal line for each charging period in Fig. 8(a), which represents the learned charging target. In Fig. 8(b), 8(d), and 8(f), the sum cost trajectory is displayed as a curve, where the sum cost at time step t is derived by the negative

reward $-r(S_t, A_t)$ in (11). Since the sum cost consists of two components, i.e., the charging cost and the cost of entering terminal states, we use the bar chart to visualize the individual costs in detail.

Fig. 8(b), 8(d), and 8(f) show that there is no cost for entering the terminal states, which means that the basic requirement of ensuring sufficient battery energy can be met by all the algorithms. A closer examination of Fig. 8(a) and 8(c) shows that the battery trajectory curve of HDDQN-HER fluctuates much more than that of DDQN-Original, which is because the high-level DDQN in the proposed algorithm can better learn the fluctuating trends of electricity prices in a full day. Therefore, the high-level agent decides to charge more when the electricity price is low in the afternoon from 14:00 to 16:00, and sell more electricity when the price is the highest, namely from 17:00 to 19:00 to minimize its charging cost. Meanwhile, DDQN-Original is unable to learn such an efficient policy, which leads to the relatively high charging cost seen in Fig. 8(d). Additionally, Fig. 8(a) shows that HDDQN-HER can realize the charging target in every charging period. Finally, as shown in Fig. 8(e), the battery SoC curve of DDQN-Low continues to remain high, since the charging target is the constant full battery capacity. Due to the cost of range anxiety, the agent tends to fully charge the EB in most charging periods, which is an essential reason why DDQN-Low has the highest charging cost. It can be observed in Fig. 8(e) that the EB is always fully charged except in the 8-th and 9-th charging periods, which means that the cost of range anxiety is always zero except in these two periods.

VIII. CONCLUSIONS

We have studied how to use HDRL techniques to learn optimal charging schedules for EBs in a system model giving full cognizance to both the charging and operating periods. An MDP model has been conceived, which has been reformulated into a two-level decision process in order to overcome the challenges of long-range multi-phase planning with sparse rewards. It has been proved that the flat policy created by superimposing the optimal high-level and low-level policies performs as well as the optimal policy of the original MDP. The HDDQN-HER algorithm has been conceived for training the two levels of agents simultaneously, in which compelling methods leveraging the idea of HER are proposed to deal with the non-stationarity encountered in training the high-level agent and to improve the sample efficiency in training the low-level agent. Finally, experiments have been performed to demonstrate that the proposed algorithm achieves better performance than the baseline algorithms.

In our future work, the HDRL framework will be extended to consider charging schedules for EB fleets, where the high-level bus scheduling decisions that are made at a coarser time-scale can be optimized jointly with the low-level charging power decisions made at a finer time-scale. Our ultimate

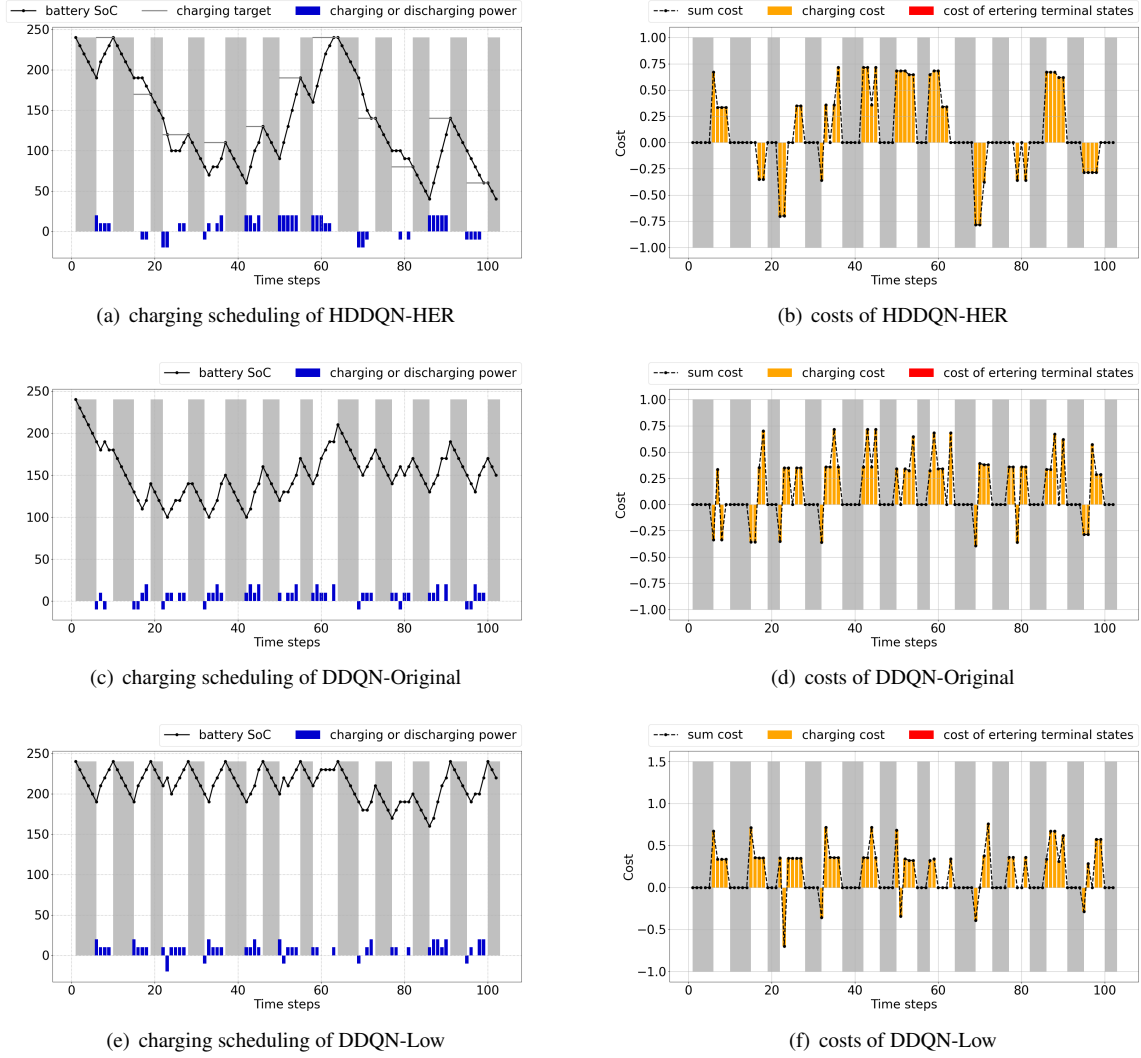


FIGURE 8: A specific test episode on an EB.

goal is to leash all solutions of the multi-component Pareto-front having a gradually increased number of parameters in the objective function. Additionally, we will integrate safe RL techniques to provide formal guarantees that the agent does not enter terminal states. This enhancement will further improve the robustness of our approach in real-world deployments. Finally, we plan to incorporate real-world EB operational data instead of simulated data in our experiments to better capture real-world uncertainties and refine policy performance under practical conditions.

APPENDIX A PROOF FOR THEOREM 1

Proof. Without loss of generality, the R.H.S. of (33) in Theorem 1 can be derived as

$$\begin{aligned}
 V_{\pi^*}(s) &\stackrel{(a)}{=} \max_{\pi} V_{\pi}(s) \stackrel{(b)}{=} \max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{T-1} r(S_t, A_t) | S_0 = s \right] \\
 &\stackrel{(c)}{=} \max_{\pi} \mathbb{E}_{\pi} \left[\sum_{k=0}^{K-1} \sum_{l=0}^{T_k^a-1} r(S_{t_k+l}, A_{t_k+l}) | S_0 = s \right], \forall s \in \mathcal{S},
 \end{aligned} \tag{35}$$

where (a) is because π^* is the optimal flat policy; (b) is derived by definition of $V_{\pi}(s)$; (c) reformulates the reward summation by grouping rewards according to periods.

When a charging period $k \in \{0, 1, \dots, K-1\}$ terminates after a random number of T_k^c time steps, we denote the SoC of the battery when following the optimal flat policy π^* as $E_{t_k+T_k^c}^*$.

In the following, we will prove Theorem 1 in **two steps**. Specifically, we decouple π^* into a high-level policy over options $\hat{\mu}$ and a low-level intra-option policy $\hat{\pi}_\omega$. Thus, the SoC levels $E_{t_k+T_k^c}^*$, $\forall k \in \{0, 1, \dots, K-1\}$ can be regarded as the charging targets prescribed by $\hat{\mu}$, i.e.,

$$\hat{\mu}(S_{t_k}) = \hat{\omega} = E_{t_k+T_k^c}^*, \forall k \in \{0, 1, \dots, K-1\}, \quad (36)$$

while the charging target $\hat{\omega}$ is fulfilled by the policy $\hat{\pi}_\omega$, i.e.,

$$\hat{\pi}_\omega(S_t) = \pi^*(S_t), \forall t \in \{t_k, \dots, t_k + T_k^c - 1\}. \quad (37)$$

In the first step, we will prove that the optimal low-level intra-option policy π_ω^* achieves the same performance as $\hat{\pi}_\omega$ does when their charging targets are both prescribed by $\hat{\mu}$ in (36), i.e.,

$$\begin{aligned} & \mathbb{E}_{\pi_\omega^*} \left[\sum_{l=0}^{T_k^c} r(S_{t_k+l}, A_{t_k+l}) | S_{t_k} = s \right] \\ &= \mathbb{E}_{\hat{\pi}_\omega} \left[\sum_{l=0}^{T_k^c} r(S_{t_k+l}, A_{t_k+l}) | S_{t_k} = s \right]. \end{aligned} \quad (38)$$

For this purpose, we adopt the proof by contradiction method and assume that (38) does not hold. Firstly, we have

$$\begin{aligned} & \mathbb{E}_{\pi_\omega^*} \left[\sum_{l=0}^{T_k^c} r(S_{t_k+l}, A_{t_k+l}) | S_{t_k} = s \right] \\ & \stackrel{(a)}{=} \mathbb{E}_{\pi_\omega^*} \left[\sum_{l=0}^{T_k^c} r^L(S_{t_k+l}, \hat{\omega}, A_{t_k+l}) | S_{t_k} = s \right] \\ & \stackrel{(b)}{=} \max_{\pi_\omega} \mathbb{E}_{\pi_\omega} \left[\sum_{l=0}^{T_k^c} r^L(S_{t_k+l}, \hat{\omega}, A_{t_k+l}) | S_{t_k} = s \right] \\ & \stackrel{(c)}{\geq} \mathbb{E}_{\hat{\pi}_\omega} \left[\sum_{l=0}^{T_k^c} r^L(S_{t_k+l}, \hat{\omega}, A_{t_k+l}) | S_{t_k} = s \right] \\ & \stackrel{(d)}{=} \mathbb{E}_{\hat{\pi}_\omega} \left[\sum_{l=0}^{T_k^c} r(S_{t_k+l}, A_{t_k+l}) | S_{t_k} = s \right], \end{aligned} \quad (39)$$

where (a) and (d) are valid, because there is no penalty for failing to realize the charging target in (26), i.e., $c^{\text{tar}}(s, \omega) = 0$ since both π_ω^* and $\hat{\pi}_\omega$ can realize the charging target $\hat{\omega}$; (b) and (c) are true according to the definition of the optimal value function in (30).

Since we assume that (38) does not hold, (c) in (39) is a strict inequality due to the property of the max operator, which means that we have

$$\begin{aligned} & \mathbb{E}_{\hat{\mu}} \mathbb{E}_{\pi_\omega^*} \left[\sum_{k=0}^{K-1} \sum_{l=0}^{T_k^c} r(S_{t_k+l}, A_{t_k+l}) | S_0 = s \right] \\ & \stackrel{(a)}{>} \mathbb{E}_{\hat{\mu}} \mathbb{E}_{\hat{\pi}_\omega} \left[\sum_{k=0}^{K-1} \sum_{l=0}^{T_k^c} r(S_{t_k+l}, A_{t_k+l}) | S_0 = s \right] \\ & \stackrel{(b)}{=} \mathbb{E}_{\pi^*} \left[\sum_{k=0}^{K-1} \sum_{l=0}^{T_k^c} r(S_{t_k+l}, A_{t_k+l}) | S_0 = s \right] \\ & \stackrel{(c)}{=} \mathbb{E}_{\pi^*} \left[\sum_{k=0}^{K-1} \sum_{l=0}^{T_k^a-1} r(S_{t_k+l}, A_{t_k+l}) | S_0 = s \right], \end{aligned} \quad (40)$$

where (a) is derived from (39) with strict inequality; (b) is due to the fact that π^* is the superimposition of $\hat{\mu}$ and $\hat{\pi}_\omega$; (c) is because of the zero rewards during the operating periods from T_k^c to $T_k^a - 1$, since there is no charging cost in the operating period. Moreover, the optimal flat policy π^* ensures that there is no penalty for entering the terminal states, i.e., $C^{\text{end}} = 0$ in (11).

Furthermore, (40) implies that the flat policy created by superimposing the policy $\hat{\mu}$ and the optimal intra-option policy π_ω^* performs better than the optimal policy π^* for the original MDP, which cannot be true by the definition of π^* . Therefore, (40) contradicts to (35), and we have thus proved that (38) holds.

In the second step, we will prove that the optimal high-level policy over options μ^* achieves the same performance as $\hat{\mu}$ does when the low-level intra-option policy is π_ω^* , i.e.,

$$\begin{aligned} & \mathbb{E}_{\mu^*} \mathbb{E}_{\pi_\omega^*} \left[\sum_{k=0}^{K-1} r^H(S_{t_k}, \omega^*) | S_0 = s \right] \\ &= \mathbb{E}_{\hat{\mu}} \mathbb{E}_{\pi_\omega^*} \left[\sum_{k=0}^{K-1} r^H(S_{t_k}, \hat{\omega}) | S_0 = s \right]. \end{aligned} \quad (41)$$

In order to prove (41), we also adopt the proof by contradiction method and assume that (41) does not hold. Thus, we

have

$$\begin{aligned}
& \mathbb{E}_{\mu^*} \mathbb{E}_{\pi_{\omega^*}^*} \left[\sum_{t=0}^{T-1} r(S_t, A_t) | S_0 = s \right] \\
& \stackrel{(a)}{=} \mathbb{E}_{\mu^*} \mathbb{E}_{\pi_{\omega^*}^*} \left[\sum_{k=0}^{K-1} r^H(S_{t_k}, \omega^*) | S_0 = s \right] \\
& \stackrel{(b)}{=} \max_{\mu} \mathbb{E}_{\mu} \mathbb{E}_{\pi_{\omega^*}^*} \left[\sum_{k=0}^{K-1} r^H(S_{t_k}, \omega) | S_0 = s \right] \\
& \stackrel{(c)}{>} \mathbb{E}_{\hat{\mu}} \mathbb{E}_{\pi_{\hat{\omega}}^*} \left[\sum_{k=0}^{K-1} r^H(S_{t_k}, \hat{\omega}) | S_0 = s \right] \\
& \stackrel{(d)}{=} \mathbb{E}_{\hat{\mu}} \mathbb{E}_{\hat{\pi}_{\hat{\omega}}^*} \left[\sum_{k=0}^{K-1} r^H(S_{t_k}, \hat{\omega}) | S_0 = s \right] \\
& \stackrel{(e)}{=} \mathbb{E}_{\pi^*} \left[\sum_{t=0}^{T-1} r(S_t, A_t) | S_0 = s \right] \stackrel{(f)}{=} V_{\pi^*}(s), \quad (42)
\end{aligned}$$

where (a) is derived by definition of the high-level reward in (19); (b) is derived by definition of the optimal value function in (23); (c) is a strict inequality due to the property of the max operator and the assumption that (41) does not hold; (d) is due to (38); (e) is because the policies $\hat{\mu}$ and $\hat{\pi}_{\hat{\omega}}^*$ superimpose the optimal flat policy π^* ; (f) is by definition of the value function in (13).

Finally, (42) implies that the flat policy created by superimposing μ^* and $\pi_{\omega^*}^*$ performs better than the optimal policy π^* for the original MDP, which cannot be true by the definition of π^* . Therefore, (42) contradicts to (35), and we have thus proved that (41) holds. Moreover, if (41) holds, (c) in (42) should be an equality. Notably, the first term in (42) corresponds to the right-hand side of (33), while the last term corresponds to its left-hand side, which means that (33) in Theorem 1 holds. \square

REFERENCES

- [1] T. Aldhanhani, A. Abraham, W. Hamidouche, and M. Shaaban, "Future trends in smart green iot: Vehicle-to-everything in the era of electric vehicles," *IEEE Open Journal of Vehicular Technology*, vol. 5, pp. 278–297, 2024.
- [2] G. Wang, X. Xie, F. Zhang, Y. Liu, and D. Zhang, "bcharge: Data-driven real-time charging scheduling for large-scale electric bus fleets," in *2018 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2018, pp. 45–55.
- [3] J. He, N. Yan, J. Zhang, T. Wang, Y.-Y. Chen, and T.-Q. Tang, "Battery electricity bus charging schedule considering bus journey's energy consumption estimation," *Transportation Research Part D: Transport and Environment*, vol. 115, p. 103587, 2023.
- [4] J. A. Manzolli, J. P. F. Trovao, and C. H. Antunes, "Electric bus coordinated charging strategy considering v2g and battery degradation," *Energy*, vol. 254, p. 124252, 2022.
- [5] I. S. Bayram and X. Shi, "Bidirectional charging hubs in the electric vehicle retail landscape: Opportunities and challenges for the u.k. case," *IEEE Open Journal of Vehicular Technology*, vol. 5, pp. 1470–1495, 2024.
- [6] Y. Liu, L. Wang, Z. Zeng, and Y. Bie, "Optimal charging plan for electric bus considering time-of-day electricity tariff," *Journal of Intelligent and Connected Vehicles*, vol. 5, no. 2, pp. 123–137, 2022.
- [7] Y. Bie, J. Ji, X. Wang, and X. Qu, "Optimization of electric bus scheduling considering stochastic volatilities in trip travel time and energy consumption," *Computer-Aided Civil and Infrastructure Engineering*, vol. 36, no. 12, pp. 1530–1548, 2021.
- [8] X. Tang, X. Lin, and F. He, "Robust scheduling strategies of electric buses under stochastic traffic conditions," *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 163–182, 2019.
- [9] H. Hu, B. Du, W. Liu, and P. Perez, "A joint optimisation model for charger locating and electric bus charging scheduling considering opportunity fast charging and uncertainties," *Transportation Research Part C: Emerging Technologies*, vol. 141, p. 103732, 2022.
- [10] Y. Zhou, H. Wang, Y. Wang, and R. Li, "Robust optimization for integrated planning of electric-bus charger deployment and charging scheduling," *Transportation Research Part D: Transport and Environment*, vol. 110, p. 103410, 2022.
- [11] K. Liu, H. Gao, Y. Wang, T. Feng, and C. Li, "Robust charging strategies for electric bus fleets under energy consumption uncertainty," *Transportation Research Part D: Transport and Environment*, vol. 104, p. 103215, 2022.
- [12] L. Lei, Y. Tan, G. Dahlenburg, W. Xiang, and K. Zheng, "Dynamic energy dispatch based on deep reinforcement learning in iot-driven smart isolated microgrids," *IEEE Internet of Things Journal*, vol. 8, no. 10, pp. 7938–7953, May 2021.
- [13] W. Chen, P. Zhuang, and H. Liang, "Reinforcement learning for smart charging of electric buses in smart grid," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.
- [14] Y. Yan, H. Wen, Y. Deng, A. H. Chow, Q. Wu, and Y.-H. Kuo, "A mixed-integer programming-based q-learning approach for electric bus scheduling with multiple termini and service routes," *Transportation Research Part C: Emerging Technologies*, vol. 162, p. 104570, 2024.
- [15] W. Wang, B. Yu, and Y. Zhou, "A real-time synchronous dispatching and recharging strategy for multi-line electric bus systems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 185, p. 103516, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1366554524001078>
- [16] X. Bi, R. Wang, H. Ye, Q. Hu, S. Bu, and E. Chung, "Real-time scheduling of electric bus flash charging at intermediate stops: A deep reinforcement learning approach," *IEEE Transactions on Transportation Electrification*, pp. 1–1, 2023.
- [17] Y. Zhou, G. P. Ong, Q. Meng, and H. Cui, "Electric bus charging facility planning with uncertainties: Model formulation and algorithm design," *Transportation Research Part C: Emerging Technologies*, vol. 150, p. 104108, 2023.
- [18] A. Curtis, M. Xin, D. Arumugam, K. Feiglis, and D. Yamins, "Flexible and efficient long-range planning through curious exploration," in *International Conference on Machine Learning*. PMLR, 2020, pp. 2238–2249.
- [19] S. Pateria, B. Subagdja, A.-h. Tan, and C. Quek, "Hierarchical reinforcement learning: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 5, jun 2021. [Online]. Available: <https://doi.org/10.1145/3453160>
- [20] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.
- [21] Y. Zhou, H. Wang, Y. Wang, B. Yu, and T. Tang, "Charging facility planning and scheduling problems for battery electric bus systems: A comprehensive review," *Transportation Research Part E: Logistics and Transportation Review*, vol. 183, p. 103463, 2024.
- [22] J. Ji, Y. Bie, and L. Wang, "Optimal electric bus fleet scheduling for a route with charging facility sharing," *Transportation Research Part C: Emerging Technologies*, vol. 147, p. 104010, 2023.
- [23] L. Zhang, S. Wang, and X. Qu, "Optimal electric bus fleet scheduling considering battery degradation and non-linear charging profile," *Transportation research. Part E, Logistics and transportation review*, vol. 154, pp. 102445–, 2021.
- [24] Z. Bao, J. Li, X. Bai, C. Xie, Z. Chen, M. Xu, W.-L. Shang, and H. Li, "An optimal charging scheduling model and algorithm for electric buses," *Applied Energy*, vol. 332, p. 120512, 2023.
- [25] Y. Zhou, Q. Meng, and G. P. Ong, "Electric bus charging scheduling for a single public transport route considering nonlinear charging profile and battery degradation effect," *Transportation Research Part B: Methodological*, vol. 159, pp. 49–75, 2022.
- [26] M. Rinaldi, E. Picarelli, A. D'Ariano, and F. Viti, "Mixed-fleet single-terminal bus scheduling problem: Modelling, solution scheme and potential applications," *Omega*, vol. 96, p. 102070, 2020.

- [27] Y. He, Z. Liu, and Z. Song, "Optimal charging scheduling and management for a fast-charging battery electric bus system," *Transportation Research Part E: Logistics and Transportation Review*, vol. 142, p. 102056, 2020.
- [28] J. Whitaker, G. Droge, M. Hansen, D. Mortensen, and J. Gunther, "A network flow approach to battery electric bus scheduling," *IEEE Transactions on Intelligent Transportation Systems*, 2023.
- [29] D.-F. Xie, Y.-P. Yu, G.-J. Zhou, X.-M. Zhao, and Y.-J. Chen, "Collaborative optimization of electric bus line scheduling with multiple charging modes," *Transportation Research Part D: Transport and Environment*, vol. 114, p. 103551, 2023.
- [30] Y. Zhou, Q. Meng, G. P. Ong, and H. Wang, "Electric bus charging scheduling on a bus network," *Transportation Research Part C: Emerging Technologies*, vol. 161, p. 104553, 2024.
- [31] A. G. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete event dynamic systems*, vol. 13, no. 1-2, pp. 41-77, 2003.
- [32] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181-211, 1999.
- [33] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of artificial intelligence research*, vol. 13, pp. 227-303, 2000.
- [34] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," *Advances in neural information processing systems*, vol. 10, 1997.
- [35] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.
- [36] M. Klissarov, P.-L. Bacon, J. Harb, and D. Precup, "Learnings options end-to-end for continuous action tasks," *arXiv preprint arXiv:1712.00004*, 2017.
- [37] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," *Advances in neural information processing systems*, vol. 29, 2016.
- [38] O. Nachum, S. S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," *Advances in neural information processing systems*, vol. 31, 2018.
- [39] A. Levy, G. Konidaris, R. Platt, and K. Saenko, "Learning multi-level hierarchies with hindsight," *arXiv preprint arXiv:1712.00948*, 2017.
- [40] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.
- [41] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.
- [42] "Guelph transit route 17 woodlawn watson," Online, 2023. [Online]. Available: <https://docs.google.com/spreadsheets/d/1c-W6M6Z6yyOpLBU05BJXBbttaozXzPTY/edit#gid=618450992>
- [43] Ameren, "Day-ahead and historical rtp/hss prices," Online, 2023. [Online]. Available: <https://www.ameren.com/account/retail-energy>



JIAJU QI received his B.S. degree from Beijing University of Posts and Telecommunications (BUPT), China, in 2017. He is currently pursuing his Ph.D. degree in Engineering at University of Guelph, Canada. His research interests include Machine Learning, Internet of Things, Reinforcement Learning, Smart Grid, and Electric Vehicle.



puting, and Smart Grid.

LEI LEI (SM'16) received the B.S. and Ph.D. degrees in telecommunications engineering from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2001 and 2006, respectively. She is currently an associate professor in the College of Engineering and Physical Sciences at the University of Guelph, Canada. Her research interests mainly lie in Machine Learning/Deep Reinforcement Learning, Internet of Things/Internet of Vehicles, Mobile Edge Com-



THORSTEINN JONSSON (THOR) is a technology leader with expertise in AI, mathematics, and physics. He has held CTO, CAIO, and AI Lead positions at several AI startups, including those he co-founded. He has mentored students and taught courses in mathematics, systems engineering, and computational modeling at the University of Iceland and the University of Guelph.

Thor's academic background includes a mathematics degree from the University of Iceland, where he developed GPU-accelerated algorithms for Open Quantum Systems simulation. He pursued further studies in Deep Learning and Generative AI at the University of Guelph and the Vector Institute. His research interests span various machine learning areas, including Semi-Supervised Learning, Generative AI, and Reinforcement Learning, with a focus on agent navigation, electric grid optimization, and sustainable transportation systems.



Lajos_Hanzo.

LAJOS HANZO (F'04) received Honorary Doctorates from the Technical University of Budapest (2009) and Edinburgh University (2015). He is a Foreign Member of the Hungarian Science-Academy, Fellow of the Royal Academy of Engineering (FREng), of the IET, of EURASIP. He holds the IEEE Edwin Howard Armstrong Award and the IEEE Eric Sumner Technical Field Award. For further details please see <http://www-mobile.ecs.soton.ac.uk>, and https://en.wikipedia.org/wiki/Lajos_Hanzo.

He coauthored 2000+ contributions at IEEE Xplore and 19 Wiley-IEEE Press monographs. Furthermore, he advised 125 PhD students and about 50 of them are now professors at various stages of their careers.