IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 22 August 2023, accepted 6 September 2023, date of publication 11 September 2023,
date of current version 19 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3313998

RESEARCH ARTICLE

An Optimization Method-Assisted Ensemble Deep Reinforcement Learning Algorithm to Solve Unit Commitment Problems

JINGTAO QIN, (Member, IEEE), YUANQI GAO, (Member, IEEE), MIKHAIL BRAGIN, (Senior Member, IEEE), AND NANPENG YU, (Senior Member, IEEE)
Department of Electrical and Computer Engineering, University of California, Riverside, Riverside, CA 92521, USA
Corresponding author: Nanpeng Yu (nyu@ece.ucr.edu)

ABSTRACT Unit commitment (UC) is a fundamental problem in the day-ahead electricity market, and it is critical to solve UC problems efficiently. Mathematical optimization techniques like dynamic programming, Lagrangian relaxation, and mixed-integer quadratic programming (MIQP) are commonly adopted for UC problems. However, the calculation time of these methods increases at an exponential rate with the number of generators and energy resources, which is still the main bottleneck in the industry. Recent advances in artificial intelligence have demonstrated the capability of reinforcement learning (RL) to solve UC problems. Unfortunately, the existing research on solving UC problems with RL suffers from the curse of dimensionality when the size of UC problems grows. To deal with these problems, we propose an optimization method-assisted ensemble deep reinforcement learning algorithm, where UC problems are formulated as a Markov Decision Process (MDP) and solved by multi-step deep Q-learning in an ensemble framework. The proposed algorithm establishes a candidate action set by solving tailored optimization problems to ensure relatively high performance and the satisfaction of operational constraints. Numerical studies on three test systems show that our algorithm outperforms the baseline RL algorithm in terms of computation efficiency and operation cost. By employing the output of our proposed algorithm as a warm start, the MIQP technique can achieve further reductions in operational costs. Furthermore, the proposed algorithm shows strong generalization capacity under unforeseen operational conditions.

INDEX TERMS Deep reinforcement learning, multi-step return, optimization methods, unit commitment.

I. INTRODUCTION

Unit commitment (UC) is a crucial decision-making tool used by Independent System Operators (ISOs) in the day-ahead electricity market. In UC problems, the optimal schedule of generators needs to be determined given the supply offers, demand bids, transmission network situations, and operational limits. The UC problems can be classified into different subgroups in a few ways [1]. With respect to security constraints, UC problems can be divided into conventional UC problems and security-constrained UC (SCUC) problems [2], [3]. According to whether uncertainty is considered and whether the probabilistic distribution of uncertain parameters is known [4], UC can be categorized into deterministic UC problems, stochastic UC problems [5], [6], and robust UC problems [7], [8], [9].

It is critical for enhancing the efficiency of the day-ahead electricity market to obtain near-optimal solutions to UC problems. The existing approaches to UC problems include heuristic algorithms [10], mathematical optimization algorithms, intelligent optimization algorithms [11], [12], and machine learning (ML) based approaches. Among these approaches, mathematical optimization algorithms including dynamic programming (DP), branch-and-cut algorithm [13], Benders decomposition [14], outer approximation [15], ordinal optimization [16], and column-and-constraint generation [17] have been widely studied in UC problems. Even though satisfactory performance is achieved by these methods, their

The associate editor coordinating the review of this manuscript and approving it for publication was Hamdi Abdi.

calculation time grows at an exponential rate with the number of energy resources. Thus, obtaining a near-optimal UC solution efficiently can be difficult when the renewable energy resources and corresponding uncertainties keep rising. To improve the performance of large-scale UC problem solvers, some researchers try to improve the tightness and compactness of the UC problem formulation as a mixed-integer programming (MIP) model [18], [19], [20]. Recently, a novel quantum distributed model is proposed to solve large-scale UC problems in a decomposition and coordination-supported framework in reference [21]. A temporal decomposition method was proposed in reference [22] which systematically decouples the long-horizon MIP problem into several sub-horizon models.

The main limitation of the aforementioned mathematical optimization algorithms used for solving UC problems is that they assume one-shot optimization where the UC problems need to be solved from scratch each time. In practice, UC problems are solved on a daily basis in the day-ahead market with small changes to the input data while the structure of the problem formulation stays the same [23]. Thus, the previous UC problems' solutions provide useful information that can be utilized to improve the solution quality of similar UC problems. Besides, these algorithms may not scale well with the increasing size of the power system. As the number of generating units, transmission lines, and load nodes grows, the computational requirements and solution time of these algorithms tend to increase significantly.

The recent advances in artificial intelligence motivate the development of machine learning-based methods to solve UC problems [24]. A series of machine learning techniques are proposed to extract valuable information from solved instances of UC problems to enhance the warm-start capabilities of MIP solvers in reference [23]. Neural networks are developed to imitate expert heuristics and speed up the branch-and-bound (B&B) algorithm, which achieves significant improvements on large-scale real-world application datasets including Electric Gird Optimization [25]. Unlike supervised learning, which requires labeled data, reinforcement learning (RL) is a mathematical tool for learning to solve sequential decision-making problems such as volt-var control problems in power distribution systems [26]. The reason why UC problems can be formulated as sequential decision-making problems is that the solution to UC problems is a sequence of generation units' operations and the current decision of units' scheduling is based on the status of units in the previous time period. In reference [27], UC problems for a system with 4 units are modeled as multi-stage decision-making tasks, and RL solutions are formulated through the pursuit method. Three RL algorithms including approximate policy iteration, tree search, and back sweep are proposed to minimize operational costs on a 12-unit system in reference [28]. The UC problem with 10 units is tackled as a multi-agent fuzzy RL task, and units play as agents to corporately reduce the overall operation cost in reference [29]. A method based on decentralized Q-learning to find a solution to UC problems on a system with up to 10 units is introduced in reference [30]. An RL-based guided tree search algorithm is developed to solve stochastic UC problems for a system with 30 generation units in reference [31], which uses a pre-trained policy to reduce the action space and designs a neural network as a binary classifier that sequentially predicts each bit in the action sequence.

Most of the existing RL-based algorithms have only been tested on small-scale UC problems because they suffer from the curse of dimensionality. Specifically, the number of states and feasible actions increases exponentially with the size of the UC problems. Besides, many operational constraints such as the transmission line capacity limit can not be strictly enforced in these RL-based algorithms. Moreover, the utilization of gradient-based training in these RL algorithms makes them susceptible to getting stuck in local optima.

To address the limitations of the existing mathematical optimization algorithms and RL algorithms, we synergistically combine mixed-integer programming with RL and propose an optimization method-assisted ensemble deep reinforcement learning algorithm to solve deterministic UC problems. The overall framework of the proposed approach is shown in Fig. 1. First, we establish a candidate action set by solving a series of simplified optimization problems to ensure that the solutions are feasible and can achieve decent performance. These candidate actions will serve as part of the inputs to the RL-based solution. Then, we design a multi-step deep Q-learning algorithm to find good sequential unit commitment decisions. By leveraging the multi-step return, the proposed algorithm explicitly accounts for the fact that the total impacts of a unit commitment decision may not instantly appear in the system operational cost and could influence the costs of many subsequent time steps. Finally, we propose an ensemble framework consisting of a group of deep Q-learning agents that are trained separately in parallel threads with different initial model parameters to find a better UC solution. This design can alleviate the problem that gradient-based training is prone to be trapped by a locally optimal solution.

The performance of our proposed algorithm, a baseline optimization method, as well as a state-of-the-art RL algorithm [31] are evaluated on three test systems. The experimental results show that our proposed algorithm identifies feasible unit commitment solutions with lower costs than both the Proximal Policy Optimization (PPO)-based guided tree search algorithm and the MIP given the same amount of computation time. The proposed algorithm can also accelerate solutions of MIP by using the results generated by our algorithm as warm starts. Moreover, additional scenario analysis demonstrates that our proposed algorithm possesses the great capability to solve emergency unit commitment problems in real time when there is a generation unit or transmission line outage.

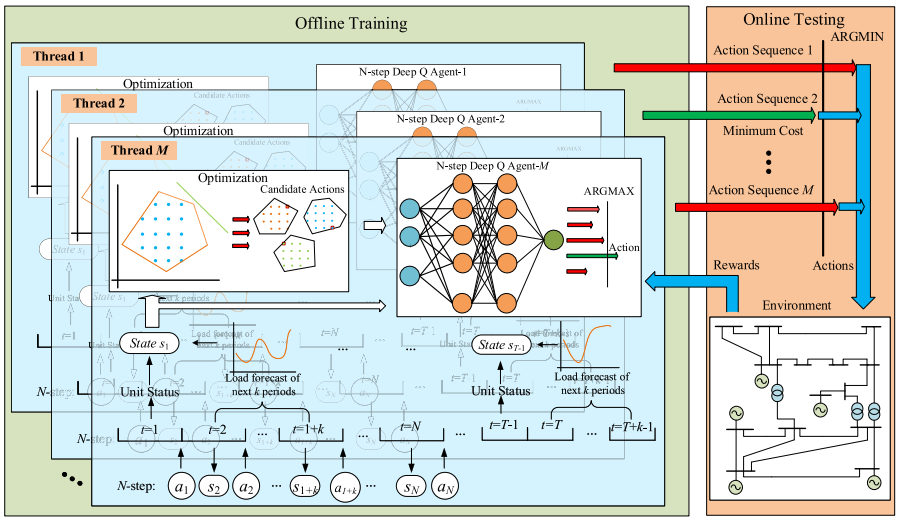


FIGURE 1. The overall framework of optimization method-assisted ensemble RL algorithm.

The unique contributions of this paper are as follows:

- This paper proposes an optimization method-assisted ensemble multi-step deep reinforcement learning algorithm that synergistically combines the merits of mixed-integer programming with reinforcement learning to accelerate the solution process of UC problems.
- The proposed algorithm establishes a candidate action set by solving a series of simplified UC problems, which ensures high-quality unit commitment solutions that satisfy operational constraints.
- The ensemble RL framework reduces the operational costs of the power system by mitigating the problem that gradient-based training of neural networks is prone to be trapped by a locally optimal solution.
- The proposed optimization method-assisted ensemble multi-step deep reinforcement learning algorithm has great capability to solve emergency unit commitment problems when there is a loss of generation units or transmission lines, which will enhance system security while simultaneously reducing operational costs.

The remainder of the paper is organized as follows: Section II gives the formulation of UC problems. Section III introduces the technical methods. Section IV discusses the experimental and algorithm setup as well as the results of numerical studies. Section V makes the conclusion.

II. PROBLEM FORMULATION

In this section, we discuss how UC problems are formulated as mixed-integer quadratic programming (MIQP) in subsection II-A, then the preliminaries of the Markov Decision Process (MDP) are provided in subsection II-B, and finally, how the UC problems are formulated as MDPs in subsection II-C.

A. FORMULATION OF UNIT COMMITMENT PROBLEMS

The objective of UC problems is to obtain the optimal commitment of generators as well as the corresponding power generating levels while minimizing the total operation cost

$$\min \sum_{t=1}^{T} \sum_{i=1}^{N} \left\{ c_i^p(t) + c_i^u(t) + c_i^d(t) \right\}, \tag{1}$$

where N is the number of units, T is the number of periods, $c_i^p(t)$, $c_i^u(t)$, $c_i^d(t)$ are the production cost, startup cost, and shutdown cost of unit i in period t, respectively. The minimization in equation (1) is subject to the following operational constraints:

Generation Capacity Constraints: Generation levels $p_i(t)$ of unit i are constrained as follows:

$$\underline{P}_i v_i(t) \le p_i(t) \le \bar{p}_i(t), \quad i = 1, \cdots, N, \ t = 1, \cdots, T, \tag{2}$$

$$0 \le \bar{p}_i(t) \le \bar{P}_i v_i(t), \quad i = 1, \cdots, N, \ t = 1, \cdots, T, \tag{3}$$

where $v_i(t)$ is the commitment status, $\bar{p}_i(t)$ is the maximum available output, $\bar{P}_i$ is the generation capacity and $\underline{P}_i$ is the minimum output of unit i at time t.

Ramp-Rate Constraints: Ramp-rate constraints require that unit i's change of power from $p_i(t-1)$ to $p_i(t)$ does not exceed RU_i while ramping up, and RD_i while ramplid down. Moreover, the ramp rate of a unit starting up cannot exceed SU_i and the ramp rate of a unit shutting down cannot exceed SD_i:

$$\begin{aligned} \bar{p}_i(t) \le p_i(t-1) + \mathrm{RU}_i v_i(t-1) + \bar{P}_i(1 - v_i(t)) \\ + \mathrm{SU}_i[v_i(t) - v_i(t-1)], \\ i = 1, \cdots, N, \ t = 1, \cdots, T, \end{aligned} \tag{4}$$

$$\begin{aligned} \bar{p}_i(t) \le \bar{P}_i v_i(t+1) + \mathrm{SD}_i[v_i(t) - v_i(t+1)], \\ i = 1, \cdots, N, \ t = 1, \cdots, T, \end{aligned} \tag{5}$$

$$p_i(t-1) \le p_i(t) + \mathrm{RD}_i v_i(t) + \mathrm{SD}_i[v_i(t-1) - v_i(t)] + \bar{P}_i[1 - v_i(t-1)], \quad i = 1, \cdots, N,\ t = 1, \cdots, T, \tag{6}$$

Minimum Up- and Down-Time Constraints: After unit i is turned on, it must stay online for UT_i time periods:

$$\sum_{t=1}^{\min(G_i,T)} [1 - v_i(t)] = 0, \quad i = 1, \cdots, N,\ t = 1, \cdots, T, \tag{7}$$

$$\sum_{n=t+1}^{\min(t+\mathrm{UT}_i,T)} v_i(n) \ge \sigma_i^u(t)[v_i(t) - v_i(t-1)], \quad i = 1, \cdots, N,\ t = G_i, \cdots, T-1, \tag{8}$$

where G_i is the number of periods during which unit i must be on or off in the beginning; $\sigma_i^u(t)$ is the number of periods that unit i must be on starting from period t, which is defined below as:

$$\sigma_i^u(t) = \begin{cases} \min(\mathrm{UT}_i, T-t), & \text{if } T > G_i, \\ 0, & \text{else}, \end{cases} \tag{9}$$

Likewise, after unit i is turned off, it must stay offline for DT_i time periods:

$$\sum_{t=1}^{\min(L_i,T)} v_i(t) = 0,\ i = 1, \cdots, N, \tag{10}$$

$$\sum_{n=t+1}^{\min(t+\mathrm{DT}_i,T)} [1 - v_i(n)] \ge \sigma_i^d(t)[v_i(t-1) - v_i(t)], \quad i = 1, \cdots, N, \quad t = L_i, \cdots, T-1, \tag{11}$$

where L_i is the number of periods during which unit i must be off in the beginning; $\sigma_i^d(t)$ is the number of periods that unit i must be off starting from period t, which is defined below as:

$$\sigma_i^d(t) = \begin{cases} \min(\mathrm{DT}_i, T-t), & \text{if } T > L_i, \\ 0, & \text{else}. \end{cases} \tag{12}$$

System Demand Constraints: The total power generated by online units, should meet the total demand for each hour t as:

$$\sum_{i=1}^{N} p_i(t) = \sum_{j=1}^{M} d_j(t), \quad t = 1, \cdots, T, \tag{13}$$

where M is the number of buses.

Spinning Reserve Requirements: The total maximum power output of online units should need the total demand as well as the spinning reserve requirement $R(t)$ of the system as time t:

$$\sum_{i=1}^{N} \bar{p}_i(t) \ge \sum_{j=1}^{M} d_j(t) + R(t), \quad t = 1, \cdots, T. \tag{14}$$

Transmission Capacity Constraints: For DC power flow, the following transmission capacity constraints apply:

$$F_l^- \le \sum_{i=1}^{N} p_i(t)\Gamma_{i,l}^P - \sum_{j=1}^{M} d_j(t)\Gamma_{j,l}^D \le F_l^+, \quad l = 1, \cdots, L,\ t = 1, \cdots, T, \tag{15}$$

where F_l^- and F_l^+ are the negative and positive power flow limit of line j; $\Gamma_{i,l}^P$ and $\Gamma_{i,l}^P$ are the power transfer distribution factor from unit i to line l.

The production cost, startup cost, and shutdown cost in equation (1) are specifically defined as follows:

1) PRODUCTION COST

Following reference [32], a quadratic production cost function is used:

$$c_i^p(t) = a_i v_i(t) + b_i p_i(t) + c_i p_i^2(t), \quad \forall i = 1, \cdots, N,\ \forall t = 1, \cdots, T, \tag{16}$$

where a_i, b_i and c_i are the coefficients.

2) STARTUP COST

A mixed-integer linear function for the stair-wise startup cost is formulated as follows:

$$c_i^u(t) \ge \mathrm{CU}_i^k \left[v_i(t) - \sum_{n=1}^{k} v_i(t-n) \right], \quad \forall i = 1, \cdots, N, \quad \forall t = 1, \cdots, T,\ \forall k = 1, \cdots, \mathrm{ND}_i, \tag{17}$$

$$c_i^u(t) \ge 0, \quad \forall i = 1, \cdots, N,\ \forall t = 1, \cdots, T, \tag{18}$$

where CU_i^k is the stair-wise startup cost of unit i in period k. ND_i is the number of intervals of the staircase startup cost function of unit i.

3) SHUTDOWN COST

The shutdown cost is defined as follows:

$$c_i^d(t) \ge \mathrm{CD}_i(v_i(t-1) - v_i(t)), \quad \forall i = 1, \cdots, N, \quad \forall t = 1, \cdots, T, \tag{19}$$

$$c_i^d(t) \ge 0, \quad \forall i = 1, \cdots, N,\ \forall t = 1, \cdots, T, \tag{20}$$

where CD_i is the shutdown cost of unit i.

B. PRELIMINARIES OF MARKOV DECISION PROCESS

As the most widely used mathematical framework to formulate sequential decision-making problems, Markov Decision Process can be defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, which consists of a state space $\mathcal{S}$, an action space $\mathcal{A}$, a state transition probability $\mathcal{P}$, a reward function $\mathcal{R}$ and a discount factor γ $(0 \le \gamma \le 1)$ [33]. This setup will allow for efficient exploitation within the Reinforcement Learning. Namely, by observing the *environment* through the above-mentioned states, the decision-maker (the *agent*) chooses an action $a_t \in \mathcal{A}$ at every time step t depending on the current state s_t, and it gains a certain reward r_{t+1}. To achieve the above, the agent first finds a policy $\pi(a|s)$ that gives the maximum anticipated discounted return $J(\pi) = \mathbf{E}[G(\tau)]$,[1] then the agent processes

[1]Here $G(\tau) = \sum_{t=0}^{T} \gamma^t r_{t+1}$, T is the length of the episode, and τ is a trajectory of states and actions.

the state by using the policy to make an appropriate decision. Subsequently, the environment shifts to the next state s_{t+1} based on $\mathcal{P}(s_{t+1}|s_t, a_t)$.

In order to demonstrate the value of states and state-action pairs given a policy π, we give the definition of two crucial value functions $v_\pi(s)$ and $q_\pi(s, a)$:

$$\begin{aligned} v_\pi(s) &= \mathbf{E}_\pi[G_t|S_t = s] \\ &= \mathbf{E}_\pi\left[\sum_{k=0}^{T} \gamma^k r_{t+k+1}|S_t = s\right], \end{aligned} \tag{21}$$

$$\begin{aligned} q_\pi(s, a) &= \mathbf{E}_\pi[G_t|S_t = s, A_t = a] \\ &= \mathbf{E}_\pi\left[\sum_{k=0}^{T} \gamma^k r_{t+k+1}|S_t = s, A_t = a\right]. \end{aligned} \tag{22}$$

We define the best policy as $\pi(a|s) = \arg\max_\pi v_\pi(s)$ for all $s \in S$ or $\pi(a|s) = \arg\max_\pi q_\pi(s, a)$ for all $s \in S$ and $a \in A(s)$.

C. FORMULATE THE UC PROBLEMS AS AN MDP

In this subsection, we construct UC problems by using MDP, while giving the definitions of the episode, state, action, and reward functions as follows.

1) EPISODE AND TIME STEPS

The episode is defined as one complete play of the RL agent interacting with the UC environment. Each operation period t is defined as a time step. Since the UC problems are solved daily in the day-ahead market, we formulate them as continuing tasks, which means an episode ends only when no feasible action can be found.

2) STATES

In order to ensure the environment is Markovian, the state at time t is defined as $s_t = (t, \boldsymbol{v}_t, \boldsymbol{p}_t, \boldsymbol{u}_t, \boldsymbol{d}_t)$, where t is the global time, $\boldsymbol{v}_t$ is a vector of the commitment status $v_i(t)$ of generator i in time t (1 if the unit is on, 0 otherwise), $\boldsymbol{p}_t$ is a vector of the power generation $p_i(t)$ of unit i in time t, $\boldsymbol{u}_t$ is a vector of the number of periods that unit i has been running or offline until time t, and the transition function of $u_i(t)$ can be formulated as equation (23):

$$u_i(t) = \begin{cases} u_i(t-1) + 1, & \text{if } v_i(t) = v_i(t-1), \\ 1, & \text{otherwise.} \end{cases} \tag{23}$$

Here $v_i(0)$ is the on/off state of unit i at the beginning of the episode, and $u_i(0)$ is the number of periods that unit i has been running or offline before the initial period of the episode. Finally, $\boldsymbol{d}_t$ is a vector $[d(t+1), d(t+2), \cdots, d(t+k)]$ of load predictions for the next k periods.

3) ACTIONS

The action a_t at time t is defined as shifting the commitment status of all generators to $\boldsymbol{v}_{t+1}$ in period $t+1$. There are numerous infeasible statuses due to the operational limitations of generators (e.g., due to minimum up- and downtime constraints). It is important to obtain all possible actions in the current state in order to avoid missing out on the best action. While infeasible actions can be filtered out, the space of feasible actions remains prohibitively large. As a result, reinforcement learning will have difficulty learning a good policy from such a large action space. This will be resolved by an optimization method to down-select candidate solutions and build a feasible action subset $\mathcal{A}_t$ in subsection III-A.

4) REWARDS

A large penalty is imposed when there is no feasible action that can be taken to prevent early termination of the episode. Accordingly, the reward function is defined as follows:

$$r_{t+1} = -\begin{cases} C_{t+1}, & \text{if } \mathcal{A}_{t+1} \neq \varnothing, \\ \zeta, & \text{if } \mathcal{A}_{t+1} = \varnothing. \end{cases} \tag{24}$$

Here C_{t+1} is the operational cost in period $t+1$ defined as:

$$C_{t+1} = \sum_{i=1}^{N} c_i^p(t+1) + \sum_{i=1}^{N} c_i^u(t+1) + \sum_{i=1}^{N} c_i^d(t+1), \tag{25}$$

where the production cost $c_i^p(t+1)$ is derived by solving a single-period economic dispatch (ED) after the commitment status $v_i(t+1)$ is obtained. Here we use $\boldsymbol{u}_t$ to directly calculate the startup cost as:

$$c_i^u(t+1) = \begin{cases} SCU_i\left[\min\{\mathrm{ND}_i, u_i(t)\}\right], & \text{if } v_i(t+1) > v_i(t) \\ 0, & \text{otherwise,} \end{cases} \tag{26}$$

where $SCU_i = \left[\mathrm{CU}_i^1, \cdots, \mathrm{CU}_i^{\mathrm{ND}_i}\right]$ is a list of the staircase startup cost of unit i, and the symbol $SCU_i[j]$ represents the j-th element of the vector SCU_i.

III. TECHNICAL METHODS

In this section, we present the proposed optimization method-assisted ensemble RL algorithm. We present a process for finding candidate actions by using optimization techniques, followed by a multi-step deep Q-learning algorithm for solving the UC problems. The ensemble framework is designed to boost performance further.

A. FINDING CANDIDATE ACTIONS USING OPTIMIZATION METHODS

The process of finding candidate actions can be divided into two parts. First, based on the current state, we solve a tailored UC problem for the next couple of periods to obtain a unit commitment schedule as the cardinal action. Then, given the cardinal action, we obtain more candidate actions by turning on or off more units based on their priority. There is a chance that certain candidate actions may be infeasible for the original Unit Commitment (UC) problem. To address this, one approach is to extend the optimization horizon of the tailored UC problem. However, it's important to note that this may lead to longer computation time.

1) FINDING THE BASE ACTION

Starting from period t, the mathematical form of a H-period UC problem can be formulated as follows:

$$\min \sum_{k=t+1}^{t+H} \sum_{i=1}^{N} \{C_i(k) + \omega_t \left(v_i(k+1) - v_i(k)\right) \rho_i, \}$$
$$s.t.\ (13) - (15) \tag{27}$$

where $C_i(k) = c_i^p(k) + c_i^u(k) + c_i^d(k)$ is the production cost of unit i in period k. ω_t is a coefficient related to t which is a positive constant when $t = 0$ and equals to zero when $t > 0$. ρ_i is the average fuel price per output power of unit i, which is given in the following formula (28):

$$\rho_i = \frac{a_i + b_i \bar{P}_i + c_i \bar{P}_i^2}{\bar{P}_i}. \tag{28}$$

After solving the UC problem above, we can obtain the unit commitment schedule of next H-period as $\boldsymbol{v}_{t+1}, \cdots, \boldsymbol{v}_{t+H}$ and we set $\boldsymbol{v}_{t+1}$ to be the cardinal action $\boldsymbol{v}_{t+1}^*$ of next period.

2) OBTAINING MORE CANDIDATE ACTIONS

Assume there are X units which are turned on or off at period $t+1$ if we take action $\boldsymbol{v}_{t+1}^*$, where $X = \sum_{i=1}^{N} |v_i^*(t+1) - v_i(t)|$. Then, instead of turning on/off X units, we turn on or off z units that have the higher priority by solving the following single-period UC problems:

$$\min \sum_{i=1}^{N} \left(v_i(t+1) - v_i(t)\right) \rho_i, \tag{29}$$
$$s.t.\ v_j(t+1) = v_j(t),\ \forall j \in \Theta_t, \tag{30}$$
$$\sum_{i=1}^{N} |v_i(t+1) - v_i(t)| = z,$$
$$(2) - (6),\ (13) - (15), \tag{31}$$

where z gradually increases from $\max(X - Y^-, 0)$ to $\min(X + Y^+, N)$. Y^- and Y^+ denote the parameters of the searching range for unit status change beyond X. Θ_t denotes the set of indexes of the units that cannot be turned on/off due to the minimum up/downtime limit at period $t+1$. Specifically, we add index i to Θ_t if $u_i(t) < 1$.

Note that here we can keep the top K best solutions to the single period UC problem, which means we can obtain $|Z| \times K$ candidate actions, $|Z|$ is the number of elements in the range of z. Finally, there are $|Z| \times K + 1$ candidate actions in the action subset $\mathcal{A}_t$. Note that K is a trade-off parameter. With larger K, we will have more candidate actions to find a better solution, but it also makes the training process more difficult to converge.

B. MULTI-STEP DEEP Q-LEARNING FOR UC PROBLEMS

For the purpose of solving MDPs with continuous state space [34], Deep Q-learning integrates the standard Q-learning with a deep neural network named deep Q network (DQN) $Q(s_t, a_t|\theta)$ to estimate the action-value function in equation (22). We use Adam gradient descent to train DQN to minimize the mean-squared temporal difference error $L(\theta)$:

$$L(\theta) = \mathbf{E}_{(s,a,r,s')\sim\mathcal{D}} \left(r + \gamma \max_{a'} Q(s', a'|\theta') - Q(s, a|\theta)\right)^2 \tag{32}$$

where $Q(s', a'|\theta')$ is the target neural network with the same structure as $Q(s_t, a_t|\theta)$. In order to make the training process stable, we update the parameters θ' of the target network from the parameters θ of $Q(s_t, a_t|\theta)$ in a periodical manner. $\mathcal{D}$ is the replay buffer to collect the transition tuples (s, a, r, s').

However, adopting DQN to solve UC problems without modifications can be inefficient, since taking an action may not only affect the next reward but also influence the rewards multiple steps later. We need several updates to propagate the reward to the related preceding states and actions [35], which makes the training process both time-consuming and tremendously sample-inefficient.

To address this issue, we use the multi-step return method [36] to update the action-value function $Q(s_t, a_t|\theta)$:

$$L(\theta) = (Q(s_t, a_t|\theta) - R(t))^2, \tag{33}$$

where $R(t)$ is defined as:

$$R(t) = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \max_{a' \in \mathcal{A}'} \gamma^n Q(s_{t+n}, a'|\theta') \tag{34}$$

Note that we make the agent-environment interact for n steps to acquire the rewards $r_{t+k}, k = 1, \ldots, n$ and the n-step next state s_{t+n}, and then calculate the n-step return.

Using the n-step return, the long-term, as well as the short-term impacts of taking an action, can be studied by propagating toward the exact reward, instead of bootstrapping from the target network $Q(s, a|\theta')$. Therefore, the learning efficiency of UC problems is prominently enhanced by applying multi-step deep Q-learning algorithm [37].

C. ENSEMBLE RL FRAMEWORK FOR UC PROBLEMS

We now present a multi-threaded ensemble reinforcement learning framework. The aim of designing this framework is to mitigate the problem of reaching a bad local optimal solution from a single random initialization of deep Q-network parameters. Specifically, we initialize the aforementioned multi-step deep Q-learning algorithm in different threads with different random seeds and run them in parallel. The pseudocode for training the ensemble multi-step deep Q-learning for UC problems is shown in algorithm 1. After the ensemble multi-step deep Q-learning algorithm is trained, the M instances of RL agents can be run in parallel during the testing phase. The multi-step deep Q-learning agent that identifies the unit commitment solution with the lowest operational cost will be selected as the final solution.

Here are some key implementation details of algorithm 1.

• Q-network structure: We adopt the feed-forward neural networks as the Q-networks, whose inputs are state-action

Algorithm 1 Training of Ensemble Multi-Step Deep Q-Learning for UC Problems

Initialize M evaluation Q-network with random parameters $\theta_1 \cdots \theta_M$
Initialize M target Q-network with parameters $\theta'_1 = \theta_1, \cdots, \theta'_M = \theta_M$
0: **for** thread $m = 1, \cdots, M$ **do**
0: Input historical data set of N_d days and set day $d = 1$
0: Initialize replay buffer $\mathcal{D}$ as a queue with a maximum
0: length of n
0: Initialize learning counter $\nu = 0$
0: **for** episode $= 1, \cdots, \Gamma$ **do**
0: Input historical load data of day d
0: Formulate initial state s_1 of day d
for $t = 1, \cdots, T$ **do**
0: Obtain candidate action set $\mathcal{A}_t$ of state s_t using optimization method.
0: With ϵ choose a random action a_t from $\mathcal{A}_t$, otherwise choose $a_t = \max_{a \in \mathcal{A}_t} Q(s_t, a|\theta'_m)$.
0: Obtain the schedule of units on next period $t+1$ based on action a_t.
0: Solve a single period ED and calculate reward r_{t+1} according to (24).
0: Calculate $\boldsymbol{u}_{t+1}$ according to (23) and then formulate the next state s_{t+1}.
0: Use optimization method to calculate $\mathcal{A}_{t+1}$.
0: Set $\varepsilon_t = 1$ if $\mathcal{A}_{t+1} = \varnothing$ else 0
0: Store $(s_t, a_t, r_{t+1}, s_{t+1}, \mathcal{A}_{t+1}, \varepsilon_t)$ in $\mathcal{D}$
0: **if** length$(\mathcal{D}) = n$ or $\varepsilon_t = 1$ **then**
0: $R = 0$ if $\varepsilon_t = 1$ else $\max_a Q(s_{t+1}, a|\theta'_m)$
0: **for** $i = t, t-1, \cdots, t -$ length$(\mathcal{D})$, **do**
0: Set $R = r_i + \gamma R$
0: Perform a gradient descent step on $(R - Q(s_i, a_i|\theta_m))^2$
0: Set $\nu = \nu + 1$
0: **if** mod$(\nu, I_{target}) = 0$ **then**
0: Update $\theta'_m = \theta_m$
0: **if** day d is over **then**
0: $d =$ mod$(d+1, N_d)$

pairs and outputs are the resulting Q value. The reason for selecting this architecture is that it can scale linearly with the number of generators.

- Episode initialization: Since UC problems are formulated as continuing tasks, which means we aim to maximize the overall reward received in all training episodes, we get the initial state of the current day from the final period of the previous day. Thus, the historical data of the next day will not be utilized for training until a policy that meets all load demands of the current day is found.
- Global Time encoding: In order to present the periodic nature of the problem, the global time step t, which varies from 0 to 23, is decomposed into two coordinates $[cos(2\pi t/24), sin(2\pi t/24)]$ [38].

IV. NUMERICAL STUDIES

A. EXPERIMENTAL AND ALGORITHM SETUP

In this subsection, we give the experimental and algorithm setups. All algorithms are executed on a server with a 32-core AMD Ryzen Threadripper 3970X 3.7GHz CPU.

1) EXPERIMENTAL SETUP

We apply the proposed method to solve a 48-hour period UC problem for the IEEE 118-bus system, IEEE 300-bus system, and the South Carolina 500-bus system [39]. The parameters of units such as minimum up and down time limit are obtained from reference [40]. The detailed experimental setup for the three testing systems can be found in our open-source repository.[2] The aforementioned minimum and maximum staircase startup cost **CU** of all units are equivalent to their hot start cost and cold start cost. Initial on/off time is the number of periods that one generator has been running or offline before the first period of the starting day. Here we give four different initial status setups to verify the generalization ability of our algorithm. The historical load data of the California Independent System Operator (CASIO) [41] from January 1, 2021, to July 5, 2021, is used and scaled to be suitable for the three testing systems. Note that we use 90 days for training, one week for validation, and two weeks for testing.

2) RL ALGORITHM SETUP

The hyperparameters of the benchmark PPO-guided tree search [31] and the proposed ensemble multi-step deep Q-learning algorithm are summarized in Tab. 1. We tune all parameters separately to achieve optimal performance.

TABLE 1. Hyperparameters of benchmark and proposed RL algorithms.

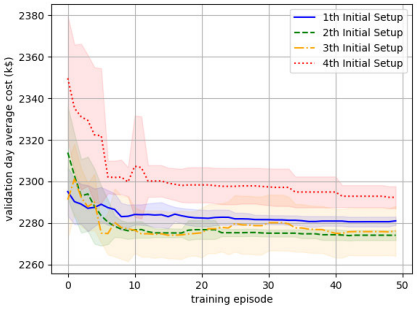
Ensemble multi-step deep-Q	Number of threads M	10
	Number of steps n	24
	Load forecast steps k	9
	Learning rate α	0.0001
	Update frequency I_{target}	60
	Greedy Range $\bar{\epsilon}$	[0.01,1.0]
	Optimization Horizon H	2
	Search Down Y^-	1
	Search Up Y^+	1
	Top K Best Actions	1
PPO-guided tree search	Actor learning rate	0.003
	Critic learning rate	0.001
	Clipping ξ	0.2
	Search Depth H	2
	Branching Threshold ρ	0.1
Shared parameters	Number of hidden units	$\{150, 150\}$
	Number of hidden layers	1
	Discount factor γ	0.99
	Number of episode	50
	Optimizer	Adam
	Activation function	Relu

B. PERFORMANCE COMPARISON

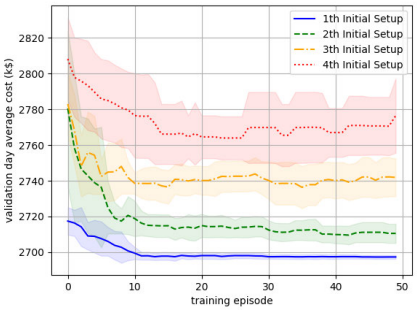
In this subsection, we compare the performance of the PPO-guided tree search, the MIQP algorithm with

[2] https://github.com/jqin020/Emsemble-Deep-RL-for-UC-problems

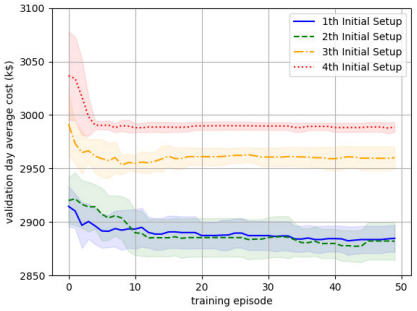
Gurobi 9.1 [42] solver, and our proposed algorithm. In all following analyses, for Gurobi we set the time limit to 10 minutes and the MIP gap to 0.01%, whichever comes first. Note that the optimization periods of MIQP are 48 hours and we obtain the operation cost of the first day from the optimization result. The mean daily operational cost of our proposed algorithm during the validation days of the three testing systems under four initial status setups are reported after every training episode in Fig. 2. The beginning commitment status of generators on the first validation day is the same during the training process. The solid curves and shaded areas represent the average values and standard deviations across different runs in 10 threads, respectively.



(a) 118-bus system



(b) 300-bus system



(c) 500-bus system

FIGURE 2. The average daily cost of validation days.

As shown in Fig. 2, the average daily costs of validation days decrease rapidly as the training continues and maintain a low level under all four initial status setups. When training processes are done, the testing days are adopted to evaluate the algorithms. For PPO-guided tree search and our proposed algorithm, we use the network parameters that yield the minimum average daily costs of the validation dataset for testing.

The average daily operation cost of testing days across four initial status setups of three testing systems are summarized in Tab. 2 to Tab. 4, respectively. The percentage variation of both the PPO-guided tree search and the proposed Ensemble N-step Q-learning from the MIQP, δ_1 and δ_2 respectively, are also given to compare their performance.

TABLE 2. Daily operation cost of the 118-bus system.

Week -Day	PPO tree search ($)	Ensemble N-step Q ($)	MIQP ($)	δ_1(%)	δ_2(%)
1-1	2,816,927	2,251,095	2,245,754	1.45	0.24
1-2	2,647,468	2,106,565	2,073,649	3.30	1.59
1-3	2,853,497	2,322,511	2,308,647	4.82	0.60
1-4	2,870,898	2,344,690	2,327,352	5.64	0.74
1-5	2,876,388	2,343,923	2,323,781	5.93	0.87
1-6	2,847,783	2,339,326	2,301,149	5.95	1.66
1-7	2,831,246	2,330,793	2,285,692	5.81	1.97
2-1	3,058,294	2,535,563	2,511,276	0.91	0.97
2-2	3,021,903	2,524,269	2,490,130	2.64	1.37
2-3	3,010,909	2,524,673	2,487,975	3.90	1.48
2-4	2,906,699	2,399,318	2,348,203	3.93	2.18
2-5	2,885,750	2,412,025	2,365,566	4.03	1.96
2-6	3,081,577	2,647,766	2,606,349	2.87	1.59
2-7	2,796,340	2,935,628	2,907,972	2.01	0.95

TABLE 3. Daily operation cost of the 300-bus system.

Week -Day	PPO-guided tree search ($)	Ensemble N-step Q ($)	MIQP ($)	δ_1(%)	δ_2(%)
1-1	2,844,044	2,736,283	2,702,568	5.23	1.25
1-2	2,548,734	2,503,574	2,484,569	2.58	0.76
1-3	2,881,019	2,769,865	2,755,564	4.55	0.52
1-4	2,894,951	2,798,182	2,780,654	4.11	0.63
1-5	2,934,550	2,798,752	2,778,608	5.61	0.72
1-6	2,902,320	2,772,518	2,746,544	5.67	0.95
1-7	2,866,755	2,761,019	2,738,949	4.67	0.81
2-1	3,124,671	3,069,440	3,011,760	3.75	1.92
2-2	3,046,178	3,003,487	2,977,668	2.30	0.87
2-3	3,054,133	3,003,764	2,972,566	2.74	1.05
2-4	2,890,670	2,852,659	2,807,185	2.97	1.62
2-5	2,908,886	2,865,494	2,831,723	2.72	1.19
2-6	3,193,841	3,152,416	3,121,763	2.31	0.98
2-7	3,552,765	3,499,896	3,471,958	2.33	0.80

TABLE 4. Daily operation cost of the 500-bus system.

Week -Day	PPO-guided tree search ($)	Ensemble N-step Q ($)	MIQP ($)	δ_1(%)	δ_2(%)
1-1	3,549,315	2,914,691	2,866,915	23.80	1.67
1-2	3,327,740	2,703,534	2,638,381	26.13	2.47
1-3	3,556,423	2,964,527	2,918,775	21.85	1.57
1-4	3,575,735	2,987,438	2,942,108	21.54	1.54
1-5	3,579,062	2,987,382	2,941,296	21.68	1.57
1-6	3,552,901	2,960,946	2,910,310	22.08	1.74
1-7	3,542,087	2,948,115	2,913,829	21.56	1.18
2-1	3,853,485	3,221,384	3,172,680	21.46	1.54
2-2	3,772,796	3,180,762	3,121,204	20.88	1.91
2-3	3,767,703	3,169,001	3,129,194	20.40	1.27
2-4	3,621,150	3,018,659	2,966,289	22.08	1.77
2-5	3,629,776	3,026,885	2,984,438	21.62	1.42
2-6	3,877,078	3,311,642	3,258,709	18.98	1.62
2-7	4,205,935	3,646,493	3,584,884	17.32	1.72

From Tab. 2 to Tab. 4 we can see that the average daily operation cost, as well as the percent variation of the proposed

algorithm from MIQP is much smaller than that of the PPO-guided tree search, while the computation time of our algorithm is shorter than that of PPO-guided tree search and MIQP. Additionally, the total computation time of testing weeks of the three testing systems is shown in Tab. 5. Here we only compare the testing time of RL-based and MIQP algorithms since the training process of RL-based algorithms can be done in an offline manner. For the 118-bus system, the training time of our proposed algorithm is approximately 2 hours and the training time of the baseline algorithm is around 9 hours. The speed-up factors are calculated by dividing the computation time of both algorithms by the computation time of MIQP. The values before the slash are the speed-up factor of the PPO tree search. Compared to the MIQP algorithms, our proposed RL-based algorithm achieves about 30 times reduction in computation time on average.

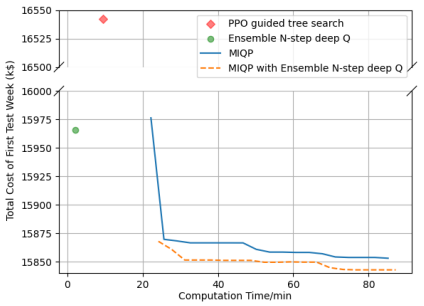
TABLE 5. Total computation time of testing weeks.

		PPO tree search (s)	Ensemble N-step Q (s)	MIQP (s)	Speed Up
118-bus	week 1	583	113	4,278	7.3/37.9
	week 2	601	122	4,283	7.1/35.1
300-bus	week 1	741	125	4,355	5.9/34.8
	week 2	756	144	4,353	5.8/30.2
500-bus	week 1	388	179	4,462	11.5/24.9
	week 2	469	167	4,451	9.5/26.7

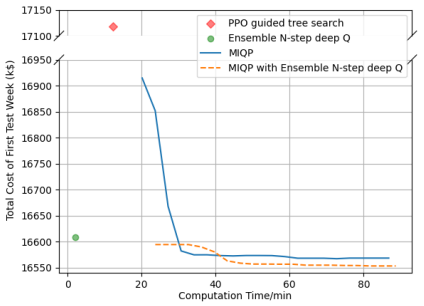
To further demonstrate the improvement of our proposed algorithm, the total operation costs of the first test week of each of the three testing systems computed by PPO-guided tree search, ensemble multi-step deep Q-learning, MIQP, and MIQP using the result of ensemble multi-step deep Q-learning as a warm start, are shown in Fig. 3. From the figure, we can see that the performance of PPO-guided tree search is close to MIQP while our proposed algorithm clearly outperforms MIQP. In other words, to identify a unit commitment solution of the same total operation cost, our proposed ensemble n-step deep Q-learning only needs a fraction of the computation time required by PPO-guided tree search and the MIQP algorithm. Given sufficient computation time, the MIQP algorithm will as expected eventually identify a solution, which has a lower operational cost than our proposed method. After using the result of our proposed algorithm as a warm start, MIQP can achieve an even lower operational cost.

C. ABLATION STUDY

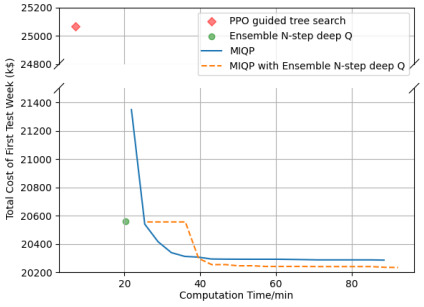
In this subsection, we study the impact of systematically removing some features from our proposed algorithm. We start by comparing the performance of using one-step return and using multi-step return during the training process under the first initial status setup. As shown in Fig. 4, the average daily costs of validation days calculated by ensemble multi-step deep Q-learning stabilize at a lower level than that of ensemble one-step deep Q-learning for all three testing systems, and the standard deviations of average daily costs



(a) 118-bus system



(b) 300-bus system



(c) 500-bus system

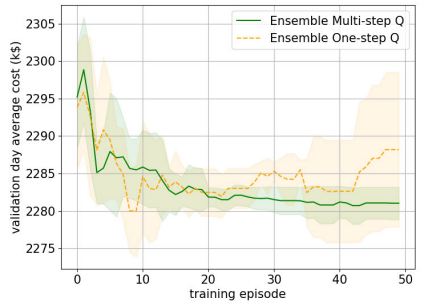
FIGURE 3. Comparison of three algorithms.

across different runs received by using multi-step return are much smaller than that of using one-step return for IEEE 118-bus, 300-bus, and the South Carolina 500-bus systems.

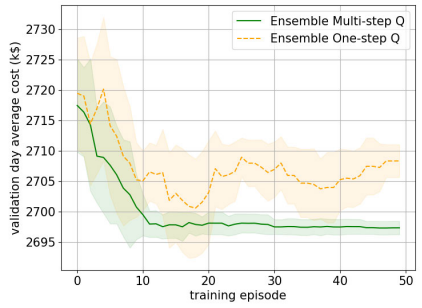
After the training process ends, we summarize the average total cost of two test weeks under the first initial status setup using different combinations in Tab. 6. Note that here we use the final parameters of the neural networks for both ensemble one-step and multi-step deep Q-learning, and we calculate the average total cost of M runs for one-step and multi-step deep-Q learning. It can be seen that the total costs of test weeks are the smallest when using a multi-step return and an ensemble framework.

TABLE 6. Total cost of test weeks using different RL techniques.

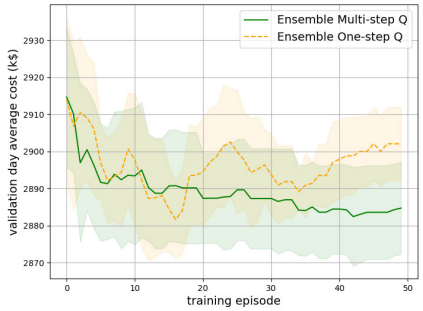
		Traditional (k\$)	Ensemble (k\$)
118-bus	One-step return	34,373	34,197
	Multi-step return	34,152	33,985
300-bus	One-step return	41,087	40,703
	Multi-step return	40,719	40,488
500-bus	One-step return	43,002	42,634
	Multi-step return	42,713	42,481



(a) 118-bus system



(b) 300-bus system



(c) 500-bus system

FIGURE 4. Comparison of one-step and multi-step return.

D. EMERGENCY UNIT COMMITMENT

In this subsection, we compare the performance of three algorithms under emergency scenarios for extended short-term unit commitment (STUC) of the South Carolina 500-bus system. When an unexpected outage occurs on a generation unit or a transmission line, the ISOs need to perform STUC immediately to obtain a near-optimal unit commitment solutions in a very short period of time. For example, the California ISO executes the STUC with a planning horizon of 18 hours and 15-minute operation interval to commit and decommit generation units incrementally.

In this experiment, we set the time limit of MIQP to 10 minutes and the MIP gap at 0.1%. Note that for our proposed algorithm and PPO-guided tree search, we set the power output of the disconnected generation unit to zero. The extended STUC is run for 7 consecutive 18-hour horizons. The average cost and computation time for the first test week when losing one unit or one transmission line are reported in Tab. 7 and Tab. 8.

As shown in the tables, our proposed algorithm yields smaller average operational costs and computation time than that of the PPO-guided tree search and MIQP. By leveraging a streamlined optimization method to identify candidate solutions and combining it with the RL algorithm, our proposed method can respond quickly to emergency events and provide a better unit commitment solution, which will enhance the system security and lower the operational cost at the same time.

TABLE 7. Average cost of emergency unit commitment when losing one unit.

Loss of Unit	PPO tree search ($)	Ensemble N-step Q ($)	MIQP ($)
Unit 3	9,169,177	8,996,524	9,113,236
Unit 13	9,173,736	8,996,524	8,996,944
Unit 35	9,173,092	8,996,528	9,045,213
Unit 46	9,169,302	8,996,524	9,019,471
Unit 80	9,166,684	8,996,524	9,032,852
Avg. Time	286.65s	33.13s	656.88s

TABLE 8. Average cost of emergency unit commitment when losing one line.

Loss of Line	PPO tree search ($)	Ensemble N-step Q ($)	MIQP ($)
Line 3	9,157,136	8,996,653	9,042,034
Line 42	9,164,030	8,996,119	9,045,970
Line 97	9,161,694	8,996,655	9,029,225
Line 151	9,171,445	8,996,526	9,058,109
Line 218	9,165,311	8,996,130	9,043,274
Avg. Time	284.58s	33.18s	655.73s

V. CONCLUSION

This paper proposes an optimization method-assisted ensemble deep reinforcement learning algorithm to accelerate the solution of unit commitment problems. We establish a candidate action set by solving simplified optimization problems. Multi-step return is used to speed up the learning process and improve the sample efficiency of the reinforcement learning agent. The proposed ensemble framework can mitigate the adverse effects that the gradient-based training could lead to a bad local optimal solution. Numerical studies show that given a time limit of solution, our algorithm can achieve a better performance than the benchmark PPO-guided tree search algorithm. Specifically, our proposed RL-based algorithm achieves an average reduction in computation time of approximately 5 times and 30 times in contrast to the PPO-guided tree search algorithm and MIQP algorithm, respectively. Besides, utilizing the results generated by our proposed algorithm as a warm start enables the MIQP technique to attain additional reductions in operational costs. Furthermore, our proposed optimization method-assisted ensemble deep reinforcement learning algorithm has a great ability to perform emergency unit commitment under unforeseen operating conditions. In the future, we plan to further improve the scalability of the proposed algorithm and tackle the security-constrained unit commitment problems on larger power systems.

ACKNOWLEDGMENT

The material is based upon work supported by the University of California Office of the President under Award Number L22CR4556.

REFERENCES

[1] I. Abdou and M. Tkiouat, "Unit commitment problem in electrical power system: A literature review," *Int. J. Electr. Comput. Eng.*, vol. 8, no. 3, p. 1357, Jun. 2018.

[2] Z. Zhang, Y. Chen, X. Liu, and W. Wang, "Two-stage robust security-constrained unit commitment model considering time autocorrelation of wind/load prediction error and outage contingency probability of units," *IEEE Access*, vol. 7, pp. 25398–25408, 2019.

[3] S. Jiang, S. Gao, G. Pan, X. Zhao, Y. Liu, Y. Guo, and S. Wang, "A novel robust security constrained unit commitment model considering HVDC regulation," *Appl. Energy*, vol. 278, Nov. 2020, Art. no. 115652.

[4] N. Yang, Z. Dong, L. Wu, L. Zhang, X. Shen, D. Chen, B. Zhu, and Y. Liu, "A comprehensive review of security-constrained unit commitment," *J. Mod. Power Syst. Clean Energy*, vol. 10, no. 3, pp. 562–576, May 2022.

[5] K. Doubleday, J. D. Lara, and B.-M. Hodge, "Investigation of stochastic unit commitment to enable advanced flexibility measures for high shares of solar PV," *Appl. Energy*, vol. 321, Sep. 2022, Art. no. 119337.

[6] C. Zhao and Y. Guan, "Data-driven stochastic unit commitment for integrating wind generation," *IEEE Trans. Power Syst.*, vol. 31, no. 4, pp. 2587–2596, Jul. 2016.

[7] Z. Shi, H. Liang, and V. Dinavahi, "Data-driven distributionally robust chance-constrained unit commitment with uncertain wind power," *IEEE Access*, vol. 7, pp. 135087–135098, 2019.

[8] Y. Wang, K. Dong, K. Zeng, X. Lan, W. Zhou, M. Yang, and W. Hao, "Robust unit commitment model based on optimal uncertainty set," *IEEE Access*, vol. 8, pp. 192787–192796, 2020.

[9] G. Zhang, F. Li, and C. Xie, "Flexible robust risk-constrained unit commitment of power system incorporating large scale wind generation and energy storage," *IEEE Access*, vol. 8, pp. 209232–209241, 2020.

[10] S. Wang, X. Xu, X. Kong, and Z. Yan, "Extended priority list and discrete heuristic search for multi-objective unit commitment," *Int. Trans. Electr. Energy Syst.*, vol. 28, no. 2, Feb. 2018, Art. no. e2486.

[11] M. Nemati, M. Braun, and S. Tenbohlen, "Optimization of unit commitment and economic dispatch in microgrids based on genetic algorithm and mixed integer linear programming," *Appl. Energy*, vol. 210, pp. 944–963, Jan. 2018.

[12] Y. Zhu and H. Gao, "Improved binary artificial fish swarm algorithm and fast constraint processing for large scale unit commitment," *IEEE Access*, vol. 8, pp. 152081–152092, 2020.

[13] Q. Gao, Z. Yang, W. Yin, W. Li, and J. Yu, "Internally induced branch-and-cut acceleration for unit commitment based on improvement of upper bound," *IEEE Trans. Power Syst.*, vol. 37, no. 3, pp. 2455–2458, May 2022.

[14] C. Liu, M. Shahidehpour, and L. Wu, "Extended benders decomposition for two-stage SCUC," *IEEE Trans. Power Syst.*, vol. 25, no. 2, pp. 1192–1194, May 2010.

[15] L. Yang, J. Jian, Z. Dong, and C. Tang, "Multi-cuts outer approximation method for unit commitment," *IEEE Trans. Power Syst.*, vol. 32, no. 2, pp. 1587–1588, Mar. 2017.

[16] H. Wu and M. Shahidehpour, "Stochastic SCUC solution with variable wind energy using constrained ordinal optimization," *IEEE Trans. Sustain. Energy*, vol. 5, no. 2, pp. 379–388, Apr. 2014.

[17] Y. An and B. Zeng, "Exploring the modeling capacity of two-stage robust optimization: Variants of robust unit commitment model," *IEEE Trans. Power Syst.*, vol. 30, no. 1, pp. 109–122, Jan. 2015.

[18] G. Morales-España, J. M. Latorre, and A. Ramos, "Tight and compact MILP formulation for the thermal unit commitment problem," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 4897–4908, Nov. 2013.

[19] S. Atakan, G. Lulli, and S. Sen, "A state transition MIP formulation for the unit commitment problem," *IEEE Trans. Power Syst.*, vol. 33, no. 1, pp. 736–748, Jan. 2018.

[20] B. Yan, P. B. Luh, T. Zheng, D. A. Schiro, M. A. Bragin, F. Zhao, J. Zhao, and I. Lelic, "A systematic formulation tightening approach for unit commitment problems," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 782–794, Jan. 2020.

[21] N. Nikmehr, P. Zhang, and M. A. Bragin, "Quantum distributed unit commitment: An application in microgrids," *IEEE Trans. Power Syst.*, vol. 37, no. 5, pp. 3592–3603, Sep. 2022.

[22] K. Kim, A. Botterud, and F. Qiu, "Temporal decomposition for improved unit commitment in power system production cost modeling," *IEEE Trans. Power Syst.*, vol. 33, no. 5, pp. 5276–5287, Sep. 2018.

[23] Á. S. Xavier, F. Qiu, and S. Ahmed, "Learning to solve large-scale security-constrained unit commitment problems," *INFORMS J. Comput.*, vol. 33, no. 2, pp. 739–756, Oct. 2020.

[24] Y. Yang and L. Wu, "Machine learning approaches to the unit commitment problem: Current trends, emerging challenges, and new strategies," *Electr. J.*, vol. 34, no. 1, Jan. 2021, Art. no. 106889.

[25] V. Nair, S. Bartunov, F. Gimeno, I. von Glehn, P. Lichocki, I. Lobov, B. O'Donoghue, N. Sonnerat, C. Tjandraatmadja, P. Wang, R. Addanki, T. Hapuarachchi, T. Keck, J. Keeling, P. Kohli, I. Ktena, Y. Li, O. Vinyals, and Y. Zwols, "Solving mixed integer programs using neural networks," 2020, *arXiv:2012.13349*.

[26] W. Wang, N. Yu, Y. Gao, and J. Shi, "Safe off-policy deep reinforcement learning algorithm for volt-VAR control in power distribution systems," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 3008–3018, Jul. 2020.

[27] E. A. Jasmin, T. P. I. Ahamed, and V. P. J. Raj, "Reinforcement learning solution for unit commitment problem through pursuit method," in *Proc. Int. Conf. Adv. Comput., Control, Telecommun. Technol.*, Dec. 2009, pp. 324–327.

[28] G. Dalal and S. Mannor, "Reinforcement learning for the unit commitment problem," in *Proc. IEEE Eindhoven PowerTech*, Jun. 2015, pp. 1–6.

[29] N. K. Navin and R. Sharma, "A fuzzy reinforcement learning approach to thermal unit commitment problem," *Neural Comput. Appl.*, vol. 31, no. 3, pp. 737–750, Mar. 2019.

[30] F. Li, J. Qin, and W. X. Zheng, "Distributed *Q*-learning-based online optimization algorithm for unit commitment and dispatch in smart grid," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 4146–4156, Sep. 2020.

[31] P. de Mars and A. O'Sullivan, "Applying reinforcement learning and tree search to the unit commitment problem," *Appl. Energy*, vol. 302, Nov. 2021, Art. no. 117519.

[32] A. J. Wood, B. F. Wollenberg, and G. B. Sheblé, *Power Generation, Operation, and Control*. Hoboken, NJ, USA: Wiley, 2013.

[33] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[34] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," in *Proc. NIPS Deep Learn. Workshop*, 2013, pp. 1–9.

[35] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. ICML*, 2016, pp. 1928–1937.

[36] C. Watkins, "Learning from delayed rewards," Ph.D. thesis, King's College, Univ. Cambridge, Cambridge, U.K., 1989.

[37] J. Qin, N. Yu, and Y. Gao, "Solving unit commitment problems with multi-step deep reinforcement learning," in *Proc. IEEE Int. Conf. Commun., Control, Comput. Technol. Smart Grids (SmartGridComm)*, Oct. 2021, pp. 140–145.

[38] Y. Gao, W. Wang, and N. Yu, "Consensus multi-agent reinforcement learning for volt-VAR control in power distribution networks," *IEEE Trans. Smart Grid*, vol. 12, no. 4, pp. 3594–3604, Jul. 2021.

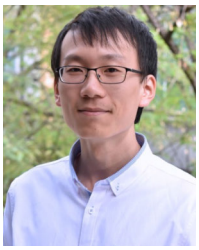
[39] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 12–19, Feb. 2011.

[40] M. Carrión and J. M. Arroyo, "A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem," *IEEE Trans. Power Syst.*, vol. 21, no. 3, pp. 1371–1378, Aug. 2006.

[41] CASIO. (2022). *California ISO Demand Forecast Website*. [Online]. Available: http://oasis.caiso.com/mrioasis/logon.do

[42] Gurobi Optimization. (2022). *Gurobi Optimizer Reference Manual*. [Online]. Available: https://www.gurobi.com

JINGTAO QIN (Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Shandong University, Jinan, China, in 2018 and 2020, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with the University of California at Riverside, Riverside, CA, USA. His research interest includes machine learning and its application in the optimization of power systems, specifically in areas such as unit commitment and distribution network reconfiguration.



YUANQI GAO (Member, IEEE) received the B.E. degree in electrical engineering from Donghua University, Shanghai, China in 2015, and the Ph.D. degree in electrical engineering from the University of California, Riverside (UCR), Riverside, CA, USA, in 2020. He was a Postdoctoral Scholar with the Department of Electrical and Computer Engineering, UCR. His research interests include big data analytics and machine learning applications in smart grids.

MIKHAIL BRAGIN (Senior Member, IEEE) is currently a Visiting Project Scientist with the Energy, Economics and Environment Research Center, University of California at Riverside, Riverside, CA, USA. His research is geared toward solving complex technical and societal challenges within smart grids, manufacturing, transportation, and healthcare. Accordingly, his research interests include operations research, mathematical optimization, artificial intelligence, machine learning, quantum computing with applications to power systems optimization, grid integration of renewables, energy-based operation optimization of distributed energy systems, decarbonization through electrification of transportation, stochastic scheduling within manufacturing systems, and pharmaceutical scheduling.

NANPENG YU (Senior Member, IEEE) received the B.S. degree in electrical engineering from Tsinghua University, Beijing, China, in 2006, and the M.S. and Ph.D. degrees in electrical engineering from Iowa State University, Ames, IA, USA, in 2007 and 2010, respectively. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of California at Riverside, Riverside, CA, USA. His current research interests include machine learning in smart grid, electricity market design and optimization, and smart energy communities. He is an Associate Editor of IEEE TRANSACTIONS ON SMART GRID and IEEE POWER ENGINEERING LETTERS.