Hierarchical Task Planning for Power Line Flow Regulation

Chenxi Wang, Youtian Du, *Member*, *IEEE*, Yanhao Huang, Yuanlin Chang, and Zihao Guo

***Abstract*—The complexity and uncertainty in power systems cause great challenges to controlling power grids. As a popular data-driven technique, deep reinforcement learning (DRL) attracts attention in the control of power grids. However, DRL has some inherent drawbacks in terms of data efficiency and explainability. This paper presents a novel hierarchical task planning (HTP) approach, bridging planning and DRL, to the task of power line flow regulation. First, we introduce a three-level task hierarchy to model the task and model the sequence of task units on each level as a task planning-Markov decision processes (TP-MDPs). Second, we model the task as a sequential decision-making problem and introduce a higher planner and a lower planner in HTP to handle different levels of task units. In addition, we introduce a two-layer knowledge graph that can update dynamically during the planning procedure to assist HTP. Experimental results conducted on the IEEE 118-bus and IEEE 300-bus systems demonstrate our HTP approach outperforms proximal policy optimization, a state-of-the-art deep reinforcement learning (DRL) approach, improving efficiency by 26.16% and 6.86% on both systems.**

***Index Terms*—Knowledge graph, power line flow regulation reinforcement learning, task planning.**

I. Introduction

To control a power system safely usually requires a sequence of operations that need to be performed without interruption. As a typical task in power systems, power line flow regulation relies heavily on power operators to dispatch power generation according to pre-determined power generation schedules and requires a number of humans with domain experience [1]. However, a massive amount of power equipment raises difficulty to traditional adjustment methods and causes power line flow regulation to be more difficult.

In recent years, deep reinforcement learning (DRL) methods have been becoming a powerful technique [2] in a wide range of applications, including video games [3] and continuous robot control [4], and obtain an impressive performance for the complex and uncertain power grid control problem [5]–[11]. Most of these works consider the power grid as a single-region system and employ DRL methods, e.g., deep Q-network (DQN) and proximal policy optimization (PPO), to solve the power grid control problem. Tang *et al*. [7] divided the power grid into multiple regions and established an online hierarchical optimization structure for the dispatch problem. However, there are still some limitations in real-world applications due to drawbacks of DRL as: 1) *Data (Interactive) efficiency* — a large quantity of interaction data between agents and environments is required to train DRL agents [12], [13]; 2) *Difficulty of reward design* — the reward function generally consists of multiple reward items which need to be carefully balanced; 3) *Lack of explainability* — introduction of black-box artificial neural networks increases the risk of decisions in the real world [14]. With the purpose of handling the above drawbacks of DRL, to combine planning and DRL is a promising framework to enhance DRL agents in many fields of applications.

Planning methods can effectively solve the problem of data efficiency by introducing a model containing prior knowledge in an entirely knowable environment [15], [16]. Faust *et al*. [15] enabled long-range navigation to control the robot under the direction of path planning with probabilistic roadmaps. Additionally, hierarchical planning structure can provide a certain degree of explainability for the planning [17], [18]. Therefore, to bridge planning and DRL with a hierarchical structure is a promising scheme to enable fast and accurate control of real-world objects. Different from existing hierarchical methods, e.g., the work [7] that introduces hierarchical methods for handling multiple regions, our work focuses on the dynamic control process of the power grid, and seeks to divide the control process into multiple fine-grained tasks and achieve automatic regulation of power lines through planning on these fine-grained tasks.

In this paper, we build a hierarchical task planning system called hierarchical task planner (HTP) for the task of power line flow regulation. We first formulate a three-level task hierarchy that consists of task, sub-task, and atomic task, and formulate a sequence of task units on each level as the task planning-Markov decision process (TP-MDP); then we introduce a higher planner and a lower planner in HTP to fuse planning and DRL agents. We also build a two-layer knowledge graph to assist the proposed HTP.

In summary, our contributions are four-fold:

1) We introduce a three-level task hierarchy, including task,

Manuscript received February 3, 2023; revised May 19, 2023; accepted July 6, 2023. Date of online publication December 28, 2023; date of current version December 24, 2023. This work was supported in part by the National Key R&D Program (2018AAA0101501) of China, and the science and technology project of SGCC (State Grid Corporation of China).

C. X. Wang, Y. T. Du (corresponding author, email: duyt@mail.xjtu.edu.cn), Y. L. Chang and Z. H. Guo are with the Ministry of Education Key Lab for Intelligent Networks and Network Security, Xi'an Jiaotong University, Xi'an 713599, China.

Y. H. Huang is with the State Key Laboratory of Power Grid Safety and Energy Conservation, China Electric Power Research Institute, Beijing 100192, China.

DOI: 10.17775/CSEEJPES.2023.00620

sub-task, and atomic task, to structure the task of power line flow regulation and model a sequence of task units on each level as the task planning-Markov decision processes (TP-MDPs).

2) We model the task as a sequential decision-making problem and propose a two-level hierarchical task planner (HTP), which consists of a higher planner and a lower planner to handle the sequence of sub-tasks and atomic tasks, respectively.

3) We introduce a two-layer knowledge graph consisting of three types of knowledge to assist HTP. Update of the knowledge graph benefits performance improvement of task planning.

4) We evaluate the proposed HTP in the task of power line flow regulation on the IEEE 118-bus and 300-bus systems. HTP improves average efficiency and control error by 26.16% and 19.17% on the IEEE 118-bus system and by 6.86% and 12.68% on the IEEE 300-bus system, compared with PPO, the state-of-the-art DRL method.

The rest of this paper is organized as follows: Section II presents a brief overview of related work. Section III describes the proposed HTP approach to the power line regulation task. Section IV provides experimental results, and Section V concludes the paper.

II. Related Work

A. Power System Operation

The power grid is a highly regulated system that must balance power supply and demand instantly. A large number of operations are necessary in power grid management and control. Regulating line flows is a vital operation for system operators. A variety of approaches have been proposed, including traditional automatic generation control (AGC) [19] and data-driven artificial intelligence methods [5]–[8]. As for emergency control of a power grid, traditional control approaches, including generation redispatch or tripping, load shedding, controlled system separation (or islanding), and dynamic braking [20], are usually executed by automatic protection systems. Recently developed machine intelligence methods, such as DRL, have the ability to solve sequential decision-making problems in real-time [21]. Dong *et al*. [19] investigated a developing adaptive emergency control scheme based on DRL and demonstrated the adaptiveness and robustness of the IEEE 39-bus system.

B. Task Planning

In general, planning is an important approach to the sequential decision-making problem, which is commonly formalized as Markov decision process (MDP) optimization [16]. As a large research field within artificial intelligence [2], planning takes a model as input and produces a policy for interacting with a modeled environment [12]. There have also been a number of works on planning methods. For combination of the task planning problem and practical mobile robot control in realistic environments, Hanheide *et al*. [22] presented a planning system named "Dora" with a structured knowledge schema, which introduces a switching planner that can perform effective planning with hypotheses. Combining planning and RL has also attracted the attention of researchers. Faust *et al*. [15] presented a hierarchical method named PRM-RL for long-range navigation task completion based on the combination of sampling-based planning and RL. RL agents perform a short-range policy, and the planner performs long-range roadmaps that can be navigated by RL agents.

The power grid is a very complex system, and control of this system can be formalized as a sequential decision-making problem and solved by a task planning system. In addition, the power grid consists of a variety of electrical quantities and runs based on physical rules, which can be formalized with knowledge graph. In this work, we propose a hierarchical task planner with assistance of knowledge graphs to perform an efficient regulation of power line flow in a power grid. This work addresses the two following issues: 1) how to model task hierarchy in the task of power line flow regulation? and 2) how to deal with uncertainty of power grids in the planning?

III. Methodology

A. Overview

This work addresses the task of power line flow regulation for power systems. Suppose we have a power grid consisting of N_{b} buses, N_{ld} loads, N_{le} power lines and N_{g} generators (one is considered as the slack node). This task is to dispatch generators to regulate line flow with the purpose of balancing loading rate of power lines. Meanwhile, we hope transmission loss on power lines is small enough and output of slack node meets the requirement.

Deep reinforcement learning is a popular learning paradigm and has been introduced in the control of power systems [5]–[11], [23]–[26]. Usually, a reward function needs to specify the goal for DRL agents. This work employs a commonly used reward function as follows:

$$r = \lambda_{\mathrm{th}} r_{\mathrm{th}} + \lambda_{\mathrm{tr}} r_{\mathrm{tr}} + \lambda_{\mathrm{sl}} r_{\mathrm{sl}} \tag{1}$$

where r_{th}, r_{tr} and r_{sl} denote rewards in terms of thermal limit, transmission loss and slack limit, respectively, λ_{th}, λ_{tr} and λ_{sl} are three coefficients to balance the three rewards.

However, to a traditional single DRL agent, it's difficult to balance weight of multiple rewards in a joint reward function. Due to the inherent disadvantage of DRL, this work introduces multiple DRL agents implemented with the PPO algorithm for control of power grids and proposes a planning algorithm to plan working of different agents. In addition, traditional planning methods require a complete state transition model of the environment [16]. However, in a power grid, power line flow is unknown before the process of power flow calculation. Therefore, how to model a transition model of a power grid is a challenge for applying planning methods.

Our main purpose is to learn a policy that can generate a sequence of predefined fine-grained tasks to complete the whole task, i.e., regulation of power line flow. However, in a complex and uncertain power system, the execution effect of a fine-grained task at a time slice is unknowable before execution, and thus, environment dynamics are unknowable to the planner. Most traditional planners [17] did not address

this uncertainty. This work introduces a task planning-MDP (TP-MDP) to formulate the problem of power line flow regulation and proposes a hierarchical task planner (HTP) for completing the task as shown in Fig. 1. Meanwhile, we introduce a multi-layer knowledge graph to assist HTP, which can be updated according to the planning procedure to model environment dynamics. In brief, the work consists of three steps: 1) selecting an initial power line flow to be adjusted, 2) repeating to plan multiple fine-grained tasks with assistance of knowledge graph until the goal is satisfied, and 3) extracting experience obtained during the planning process to update the knowledge graph.

B. Task Planning-Markov Decision Process

Control of a power grid can be considered a sequential decision-making problem and is formulated as a TP-MDP with a tuple $(\mathcal{S}, \mathcal{A}, R, P, I, G)$.

$\mathcal{S}$ denotes state space, being the state at time t. We define $\boldsymbol{S}_t$ as follows:

$$\boldsymbol{S}_t = (\boldsymbol{S}_{\text{power},t}, \boldsymbol{S}_{\text{plan},t}) \tag{2}$$

In TP-MDPs, $\boldsymbol{S}_{\text{power},t}$ is formulated as a vector representing electrical quantities of a power grid to describe its state and defined as follows:

$$\boldsymbol{S}_{\text{power},t} = \{L_{1:N_{\text{le}},t}, P_{1:N_{\text{ld}},t}, l_{k,t}, P_{\text{s},t}\} \tag{3}$$

where $L_{i,t}$ denotes loading rate of line i, $P_{j,t}$ is active power output of generator j, $l_{k,t}$ is active power consumed by load k and $P_{\text{s},t}$ denotes active power of slack node outputs. $\boldsymbol{S}_{\text{plan},t}$ indicates state of task planning and can be represented as a set of assignments of values to variables, including execution effects of each fine-grained task, which describe status of planning procedure. We demonstrate a typical $\boldsymbol{S}_{\text{plan}}$ as:

$$\boldsymbol{S}_{\text{plan},t} = \{v_1 = e_1, v_2 = e_2, \cdots\},\ v_1, v_2, \cdots \in \mathcal{V} \tag{4}$$

where $\mathcal{V}$ is variable space, $v_i \in \mathcal{V}$ and e_i are variable and value, respectively.

$\mathcal{A}$ represents a set of actions, each specifying a fine-grained task executed on $\boldsymbol{S}_t$. Each fine-grained task has two types of properties: precondition $\text{pre}(\text{task})$ and execution effect $\text{eff}(\text{task})$. Precondition and effect are also sets of assignments of values to variables and have the same form as $\boldsymbol{S}_{\text{plan}}$. A fine-grained task can be executed only if its precondition is satisfied. A fine-grained task may have multiple effects, and different fine-grained tasks may have the same preconditions. In addition, some fine-grained tasks perform operations to the power grid, and then $\boldsymbol{S}_{\text{power},t}$ will change accordingly.

R denotes the reward function in planning and is defined as a cost function $C : A \mapsto \mathbb{R}^+$ for an action. Each fine-grained task at t has an execution cost, which represents time it takes to execute the task. Therefore, reward R_t at t is defined as: $R_t = -c_t$.

P denotes state transition. When an action A_t is taken on $\boldsymbol{S}_t$, the system will find itself a new plan state $\boldsymbol{S}_{\text{plan},t+1}$ according to execution effect of A_t and a new $\boldsymbol{S}_t \in \mathcal{S}$ power state $\boldsymbol{S}_{\text{power},t+1}$ after power flow calculation.

$\boldsymbol{S}_{\text{init}}$ indicates initial state of the task of power line flow regulation. An initial state $\boldsymbol{S}_{\text{init}} \in \boldsymbol{S}_{\text{plan}}$ will be specified at each start of task planning.

$\boldsymbol{S}_{\text{goal}}$ represents goal state of the task of power line flow regulation. Goal state is terminated state in TP-MDPs, that is, task planning will be finished when it is reached.

If a task is performed in T steps, return of task planning can be defined as follows:

$$G = \sum_{t=0}^{T-1} \gamma^t (R_t) \tag{5}$$

where γ is discounting factor and T is terminate step. To solve a sequential task planning problem formulated by the TP-MDP, we need to learn a policy π to maximize return G.

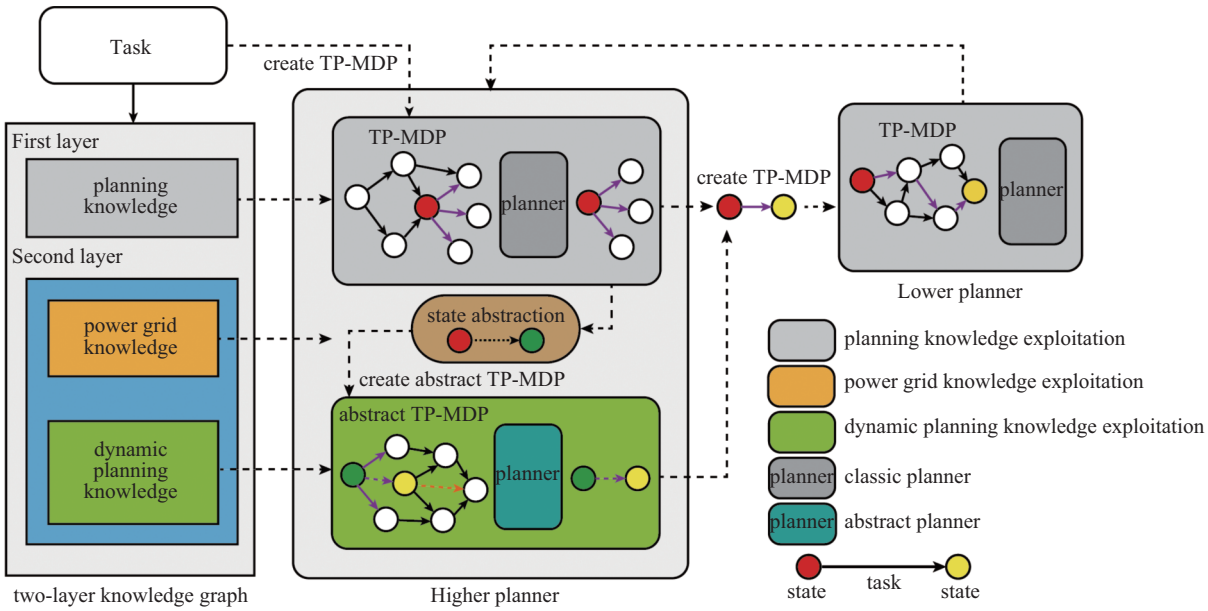

Fig. 1. Overview of the hierarchical task planner and the knowledge graph.

C. Hierarchical Task Planner Framework

1) Task Model

In this work, we model a task using a hierarchical structure with three layers: a) *task* (τ), which refers to large-scale tasks with an independent purpose; b) *sub-task* (τ_{st}), which refers to small-scale and usually reusable tasks contained in a task τ with explicit and complete functionality. A sub-task is a subdivision of a task τ with respect to specific functionality and may have uncertain or unknown execution effects; c) *atomic-task* (τ_{at}), which refers to the most fine-grained tasks that cannot be subdivided and can be directly executed for completing a sub-task. Sub-task and atomic-task can be considered as a fine-grained task mentioned in Section III-B. We formulate the three layers of tasks as follows.

Task:

$$\tau(\boldsymbol{S}_{\mathrm{init}}, \boldsymbol{S}_{\mathrm{goal}}),\ \tau \in \mathcal{T}, \boldsymbol{S}_{\mathrm{init}}, \boldsymbol{S}_{\mathrm{goal}} \in \mathcal{S} \tag{6}$$

A task $\tau \in \mathcal{T}$ specifies an initial state $\boldsymbol{S}_{\mathrm{init}}$ and a goal state $\boldsymbol{S}_{\mathrm{goal}}$. Each task creates a TP-MDP for the planner to solve.

Sub-task:

$$\tau_{\mathrm{st}}(\mathrm{pre}(\tau_{\mathrm{st}}), \mathrm{eff}(\tau_{\mathrm{st}}), \boldsymbol{S}_{\mathrm{init}}, \boldsymbol{S}_{\mathrm{goal}}),\ \tau_{\mathrm{st}} \in \mathcal{T}_{\mathrm{st}}, \boldsymbol{S}_{\mathrm{init}}, \boldsymbol{S}_{\mathrm{goal}} \in S \tag{7}$$

A sub-task $\tau_{\mathrm{st}} \in \mathcal{T}_{\mathrm{st}}$ consists of four terms: precondition $\mathrm{pre}(\tau_{\mathrm{st}})$, effect $\mathrm{eff}(\tau_{\mathrm{st}})$, initial state $\boldsymbol{S}_{\mathrm{init}}$ and goal state $\boldsymbol{S}_{\mathrm{goal}}$. Each sub-task also creates a TP-MDP for planner to solve.

Atomic task:

$$\tau_{\mathrm{at}}(\mathrm{pre}(\tau_{\mathrm{at}}), \mathrm{eff}(\tau_{\mathrm{at}}))\ \tau_{\mathrm{at}} \in \mathcal{T}_{\mathrm{at}} \tag{8}$$

An atomic task τ_{at} in atomic task space $\mathcal{T}_{\mathrm{at}}$ consists of precondition $\mathrm{pre}(\tau_{\mathrm{at}})$ and effect $\mathrm{eff}(\tau_{\mathrm{at}})$. Atomic task is the most fine-grained unit that a planner can execute, and every atomic task has a determinate execution effect on $\boldsymbol{S}_{\mathrm{plan}}$.

For the task of power line flow regulation, we demonstrate task hierarchy as shown in Fig. 2 and show details in Table I. We divide the task of power line flow regulation into 6 sub-tasks, i.e., "data prepare", 3 DRL regulation sub-tasks ("thermal limit", "slack limit" and "transmission loss"), "data analysis" and "output". Each sub-task can be subdivided into 1 or 2 atomic tasks, where "thermal limit", "slack limit" and "transmission loss" are performed with 3 DRL agents that can control power line flow. Other sub-tasks and atomic tasks are auxiliary fine-grained tasks required to complete the whole task of power line flow regulation.

2) Task Planning System

We propose HTP as illustrated in Fig. 1. This framework consists of two modules: 1) a two-level planner consisting of a higher planner π_{H} and a lower planner π_{L}, and 2) a two-layer knowledge graph, called power system domain knowledge graph $\mathcal{K}$, to assist the two-level planner.

First layer of $\mathcal{K}$, called planning knowledge graph K_{p}, stores task-related knowledge irrelevant to a specific power grid; second layer, including a power grid knowledge graph K_{d} and a dynamic planning knowledge graph K_{dp}, stores the knowledge of a specific power grid that needs to be controlled. K_{p} and K_{dp} both store preconditions and execution effects of sub-task and atomic task, which can be used to model

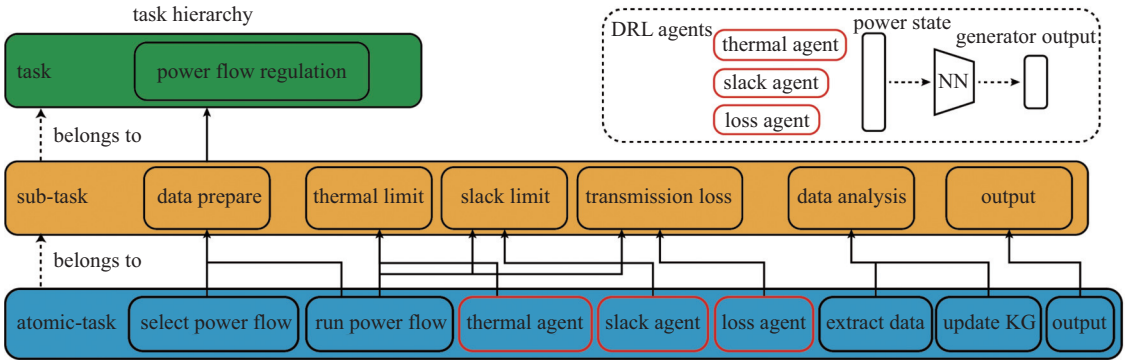

Fig. 2. The task hierarchy for the task of power line flow regulation.

TABLE I
THE DESCRIPTION OF TASK UNITS

Hierarchy	Task name	Task description
Task	power line flow regulation	Regulate the power line flow with multiple DRL agents
Sub-task	data prepare	Select the base power line flow
	thermal limit	Regulate the power line flow to control the power lines' loading rate
	slack limit	Regulate the power line flow to control the slack node output
	transmission loss	Regulate the power line flow to minimize the transmission loss
	data analysis	Analysis knowledge and update knowledge graph in the planning procedure
	output	Output control actions
Atomic-task	select power flow	Select a base power flow
	power flow calculation	Run the power flow calculation
	thermal agent	Call the thermal limit DRL agent to output the control action to the power grid
	slack agent	Call the slack limit DRL agent to output the control action to the power grid
	transmission agent	Call the transmission loss DRL agent to output the control action to the power grid
	extract data	Extract the knowledge data from the planning procedure
	update KG	Update the K_{dp} with knowledge data
	output	Output all control actions to the power grid

state transition of the power gird environment. The difference between K_{p} and K_{dp} is that the former is a fixed knowledge graph and models state transition of the environment on $\boldsymbol{S}_{\text{plan}}$, while the latter can be updated with execution effects, which are uncertain or unknown before the planning process, on $\boldsymbol{S}_{\text{power}}$ of sub-tasks. Update of K_{dp} can make state transition model of environment more accurate.

To handle uncertainty of the effect of a sub-task on $\boldsymbol{S}_{\text{power}}$, we present a two-level planner consisting of a higher planner working on sub-task level and a lower planner on atomic-task level. Action of higher planner π_{H} is the sub-task to be executed in next step, where a sub-task has its initial state and goal state. Therefore, we can define a new TP-MDP on the atomic task level with initial state and goal state specified by the sub-task. Then, lower planner π_{L} will get involved to solve the atomic task level TP-MDP and accomplish this sub-task output by the higher planner π_{H}. Action of π_{L} is the atomic task to be executed, and a sequence of atomic tasks will be generated and executed sequentially until goal state of the sub-task is reached. Two levels of planners will perform alternately until goal state of the whole task of power line flow regulation is reached.

D. Two-level Planner

1) Higher Planner

Higher planner π_{H} is a switching planner, which consists of a classic planner π_{c}, an abstract planner π_{a} and a state abstraction module. Classic planner π_{c} works for sub-tasks that have a determinate effect; π_{a} works for sub-tasks that have an indeterminate effect on $\boldsymbol{S}_{\text{power}}$; state abstraction module transforms power state $\boldsymbol{S}_{\text{power}}$ to an abstract power state $\boldsymbol{S}_{\text{abstract}}$. Both classic planner and abstract planner can output the next sub-task to be executed:

$$A_t \leftarrow \pi_{\mathrm{c}}(\boldsymbol{S}_{\text{plan},t}) \tag{9}$$

$$A_t \leftarrow \pi_{\mathrm{a}}(\boldsymbol{S}_{\text{abstract},t}, \boldsymbol{S}_{\text{plan},t}) \tag{10}$$

Classic planner. Classic planner π_{c} works on task planning state $\boldsymbol{S}_{\text{plan}}$. When a task τ is required to be completed, corresponding $\boldsymbol{S}_{\text{init}}$ and $\boldsymbol{S}_{\text{goal}}$ are then specified and a TP-MDP will be established. π_{c} will work to determine the sub-task τ_{st} that should be executed in the next step if this sub-task does not change power state $\boldsymbol{S}_{\text{power}}$ and its effect depends only on $\boldsymbol{S}_{\text{plan}}$.

State abstraction module. When preconditions of more than one sub-task are satisfied and their effects can change $\boldsymbol{S}_{\text{power}}$, we need to predict effects of each sub-task on $\boldsymbol{S}_{\text{power}}$ and make planning based on predicted effects. However, $\boldsymbol{S}_{\text{power}}$ is a numeric vector and lacks semantic meaning, so it is difficult to model transition of the state $\boldsymbol{S}_{\text{power}}$. As a solution, we transform numerical $\boldsymbol{S}_{\text{power}}$ vector into an abstract state $\boldsymbol{S}_{\text{abstract}}$ expected to possess the semantic meaning and then predict transition of abstract states. We introduce a state abstraction module to map power state to an abstract power state based on power grid knowledge graph:

$$\boldsymbol{S}_{\text{abstract}} \leftarrow \text{stateAbstraction}(\boldsymbol{S}_{\text{power}}, K_{\mathrm{d}}) \tag{11}$$

Knowledge power grid knowledge graph specifies abstract information we focus on in the task. State abstraction module can build an abstract state $\boldsymbol{S}_{\text{abstract}}$, consisting of multiple preconditions and execution effect pairs on abstract state, by extracting corresponding information from $\boldsymbol{S}_{\text{power}}$. In the dynamic planning knowledge graph, we make hypothesis about effect of each sub-task execution on abstract state, and then we can predict the transition model of abstract states and build an abstract TP-MDP regarding abstract state.

Abstract planner. Abstract planner π_{a} works on abstract power state $\boldsymbol{S}_{\text{abstract}}$. We define a TP-MDP, called abstract TP-MDP, regarding the abstract state, which will be established with the initial state specified by current $\boldsymbol{S}_{\text{abstract}}$ and goal state specified by K_{d}. State transition on $\boldsymbol{S}_{\text{abstract}}$ can be modeled by K_{dp}, and we have an abstract TP-MDP that can control power line flow from current $\boldsymbol{S}_{\text{abstract}}$ to goal state. Abstract planner π_{a} will solve abstract TP-MDP and make planning to determine which sub-task should be executed in the next step.

2) Lower Planner

After higher planner chooses a sub-task to be executed, an atomic-level TP-MDP will be initialized while specifying corresponding initial state $\boldsymbol{S}_{\text{init}}$ and goal state $\boldsymbol{S}_{\text{goal}}$ according to the sub-task, and determining state transition by K_{p}. A classic planner π_{c} is employed as lower planner to solve atomic-level TP-MDP.

3) Planning Algorithm

We implement both classic and abstract planners based on Monte-Carlo tree search (MCTS), a widely used heuristic search algorithm used in many applications [27], [28].

When a task τ is specified, the task planning system will create a TP-MDP on planning state and trigger a higher planner π_{H}. When a sub-task is determined with π_{H}, an atomic-level TP-MDP will be created, and the lower planner will start to make planning at the atomic level. When a sub-task is completed by the lower planner, π_{H} will determine the next sub-task until the whole task τ is completed. When more than one sub-task is specified, the abstract planner π_{a} belonging to the higher planner π_{H} will interpose. First, a state abstraction module will map power state $\boldsymbol{S}_{\text{power}}$ to an abstract state $\boldsymbol{S}_{\text{abstract}}$ and an abstract TP-MDP based on $\boldsymbol{S}_{\text{abstract}}$ will be created, where the state transition in the abstract TP-MDP is approximated by K_{dp}. Then, π_{a} will determine which sub-task needs to be executed. The planning procedure is illustrated in Algorithm 1.

E. Power System Domain Knowledge Graph

Knowledge graph is a semantic knowledge base that structurally describes relations between entities with a graph model. A knowledge graph is a collection of triplets (h_1, r, h_2), where h_1 and h_2 are two entities and r denotes the relation between them.

In the power system domain, operations of a power grid require a variety of expert experiences and rules. We introduce a two-layer knowledge graph to model knowledge and experience of power system domain. Fig. 3 illustrates a part of the knowledge graph used in this work. Fig. 3(a) depicts a representative planning knowledge graph in this study, where red dashed lines with arrows indicate relations across different levels in task hierarchy, black lines with arrows illustrate initial

Algorithm 1: HTP Planning Procedure

Input: Task τ;
create TP-MDP base on the task τ;
repeat
  Higher planner outputs a sub-task τ_{st} with the classic planner via (9);
  if *the classic planner is failed* **then**
    Generate abstract state $\boldsymbol{S}_{\text{abstract}}$ via (11);
    Create the abstract TP-MDP;
    Higher planner outputs a sub-task τ_{st} with classic planner via (10);
  end
  repeat
    Create an atomic-level TP-MDP for the sub-task τ_{st};
    Lower planner outputs an atomic task τ_{at} with the classic planner via (9);
    Execute the atomic task τ_{at};
  until *the sub-task* τ_{st} *is completed*;
until *the task* τ *is completed*;
return

states and goal states of predefined task and each sub-task, as well as preconditions and execution effects of each sub-task and atomic task. Fig. 3(b) shows power grid knowledge graph, which contains goal and abstract state of predefined task, i.e., power line flow regulation. Fig. 3(c) illustrates a typical knowledge triplet in dynamic planning knowledge graph, which can be updated during the planning process.

1) Planning Knowledge Graph

Planning knowledge graph K_{p} models task hierarchy, and preconditions/effects of sub-tasks and atomic task. Typical triplets in knowledge graph K_{p} are shown as follows:

$$\begin{aligned}&(\tau_{\mathrm{st}}, \text{has-effect}, \mathrm{eff}(\tau_{\mathrm{st}})) \in K_{\mathrm{p}}\\&(\tau_{\mathrm{st}}, \text{has-precondition}, \mathrm{pre}(\tau_{\mathrm{st}})) \in K_{\mathrm{p}}\end{aligned} \tag{12}$$

which denote sub-task τ_{st} has an effect $\mathrm{eff}(\tau_{\mathrm{st}})$ and a precondition $\mathrm{pre}(\tau_{\mathrm{st}})$.

2) Power Grid Knowledge Graph

For a given power grid, power grid knowledge graph K_{d} specifies goal of power line flow regulation and defines elements an abstract state consists of (cf. Fig. 3). Typical triplets in knowledge graph K_{d} are shown as follows:

$$\begin{aligned}&(\text{power grid}, \text{has-goal}, \text{goal}) \in K_{\mathrm{d}}\\&(\text{power grid}, \text{has-abstract}, \text{abstract}) \in K_{\mathrm{d}}\end{aligned} \tag{13}$$

We then extract the value of each element in abstract state from power state $\boldsymbol{S}_{\text{power}}$ and then obtain abstract state $\boldsymbol{S}_{\text{abstract}}$.

3) Dynamic Planning Knowledge Graph

We introduce a dynamic planning knowledge graph to continuously update the experience in the planning process. Dynamic planning knowledge graph K_{dp} stores the effects and preconditions of those sub-tasks that can change power state. Typical triplets in knowledge graph K_{dp} are shown as follows:

$$\begin{aligned}&(\tau_{\mathrm{st}}, \text{has-precondition}, \mathrm{pre}(\tau_{\mathrm{st}})) \in K_{\mathrm{dp}}\\&(\tau_{\mathrm{st}}, \text{has-effect}, \mathrm{eff}(\tau_{\mathrm{st}})) \in K_{\mathrm{dp}}\end{aligned} \tag{14}$$

which denote that sub-task τ_{st} has an effect $\mathrm{eff}(\tau_{\mathrm{st}})$ and a precondition $\mathrm{pre}(\tau_{\mathrm{st}})$. In this work, effects and preconditions in K_{dp} are specified for elements of abstract states defined in K_{d}, and thus $\mathrm{eff}(\tau_{\mathrm{st}})$ and $\mathrm{pre}(\tau_{\mathrm{st}})$ have the same form as $\boldsymbol{S}_{\text{abstract}}$.

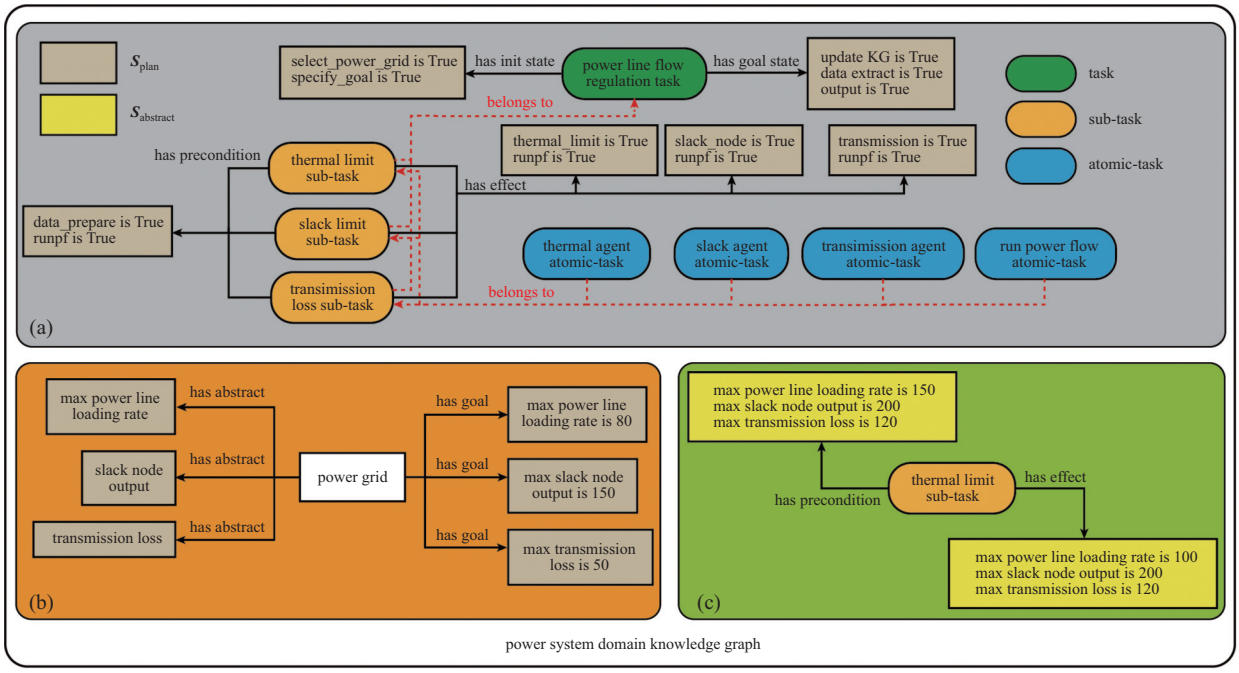

Fig. 3. Part of the power system domain knowledge graph in this work. (a) Planning knowledge graph K_{p}. (b) Power grid knowledge graph K_{d}. (c) Dynamic planning knowledge graph K_{dp}.

IV. Experimental Results and Analysis

A. Experimental Setting

For the task of power line flow regulation, we set a goal that meets the following conditions: 1) loading rate of each power line must be maintained within thermal limit; 2) power output of slack node should not exceed limitation; 3) transmission losses should be minimized.

We define the three reward terms in (1) as follows:

$$r_{\mathrm{th}} = 1 - \max_{i=1,2,\cdots,173} (L_{i,t})^2 \tag{15}$$

$$r_{\mathrm{tr}} = \sum_{i=1}^{173} (1 - (L_{i,t})^2) \tag{16}$$

$$r_{\mathrm{sl}} = \begin{cases} 1 - \frac{P_{\mathrm{s},t}}{P_{\max}}, & P_{\mathrm{s},t} \in (P_{\max}, +\infty) \\ 0, & P_{\mathrm{s},t} \in [P_{\min}, P_{\max}] \\ \frac{P_{\mathrm{s},t}}{P_{\min}} - 1, & P_{\mathrm{s},t} \in (-\infty, P_{\min}) \end{cases} \tag{17}$$

where $L_{i,t}$ is loading rate of line i; $P_{\mathrm{s},t}$ is active power generated by slack node, $P_{\max}$ and $P_{\min}$ are maximum and minimum limit of active powers of slack node, respectively. We also define actions to regulate power line flow in the following form:

$$A = (\Delta P_{1,t}, \Delta P_{2,t}, \cdots, \Delta P_{53,t}) \tag{18}$$

where $\Delta P_{j,t}$ is active power increment of generator j at time t. Active power outputs of generators are then set to $P_{j,t} + \Delta P_{j,t}$, $j = 1, 2, \cdots, 53$.

In experiments, we employ proximal policy optimization (PPO) [4], soft actor-critic (SAC) [29], and twin delayed deep deterministic policy gradient (TD3) [30] as the compared methods, which are denoted by RL-PPO, RL-SAC, and RL-TD3, respectively. In implementation of compared methods, we set all coefficients in (1) equal to 1, which means each DRL agent learned by jointly optimizing the three objectives in (1).

Each actor and critic network in PPO, SAC, and TD3 are implemented by 3 fully connected hidden layers with 512 ReLU neurons. In the actor network of all compared methods, input is a vector representing state of the power grid and output is action vector. In the critic network of SAC and TD3, input is the concatenation of a state vector and an action vector, output is state-action value of input. For PPO, input of critic network is a state vector and output is state value of input. In the training process, learning rate is set to 10^{-4} for critic network and 10^{-5} for actor network. Discounting factor is set to 0.95 and batch size in SAC and TD3 is set to 128.

Based on evaluation of the three compared methods, we found the PPO agent can obtain the best performance. In our approach HTP, we employ PPO as the base algorithm for executing sub-tasks.

Our approach (HTP): We train 3 PPO DRL agents separately, each for an objective in (1) by setting the coefficient of the corresponding objective to 1 and other two to 0. Three DRL agents can output actions to regulate power line flow corresponding to three atomic tasks, i.e., thermal agent, slack agent and transmission agent as shown in Table I.

B. A Case of Knowledge Graph

There are a variety of established rules and prior knowledge for power line flow regulation in the power system domain. We formulate these rules and knowledge into triplets and structure them into K_{p} and K_{d}.

Planning knowledge graph: In the task of power line flow regulation, K_{p} stores preconditions and effects of sub-tasks and atomic tasks. We demonstrate a part of K_{p} as shown in Fig. 3(a).

Power grid knowledge graph: We store the goal of the task in K_{d} in terms of three following items: 1) goal for max loading rate of all power lines; 2) goal for power output by slack node; and 3) goal for transmission loss in the power grid. K_{d} is utilized to map power state into an abstract power state. We demonstrate a part of K_{d} in Fig. 3(b).

Dynamic planning knowledge graph: Before task planning, the effect of each sub-task regulating the power line flow (i.e., involving a DRL agent in the execution) is unknown. We may initialize K_{dp} by assigning an assumption about execution effects for each of the three sub-tasks. As task planning procedure progresses, more knowledge of the effects is extracted, and execution effects can be estimated more accurately. A part of K_{dp} is shown in Fig. 3(c).

C. Results of Planning on IEEE 118-bus System

We employ the IEEE 118-bus system as study case as shown in Fig. 4, where red squares enclosing the generator indicate renewable units. The system consists of 54 generators including 28 renewable units, 173 transmission lines and 99 loads, where renewable units has variable output power. We design a total of 9 patterns regarding loads and renewable units. We totally collect 60,480 samples as a dataset, 57,600 for training and the rest 2,880 for test.

We designate 3 different goals for the task of power line flow regulation. Goal 1—max loading percent: 100%; max slack node output: 150 MW; max transmission losses: 50 MW. Goal 2—max loading percent: 80%; max slack node output: 65 MW; max transmission losses: 40 MW. Goal 3—max loading percent: 80%; max slack node output: 55 MW; max transmission losses: 35 MW. From goal 1 to 3, difficulty increases to achieve successful regulation.

In the task of the power flow regulation, we do not have a strict limit in terms of computational time. We are more concerned about number of control steps required and regulation accuracy. Therefore, we design two evaluation metrics as follows:

1) Control steps: We define control steps as the steps an agent takes to reach the goal of regulation. Control steps can intuitively reflect efficiency of regulation.

2) Control error: We define control error as follows:

$$E = \frac{e(l_{\mathrm{th}}, g_{\mathrm{th}})}{g_{\mathrm{th}}} + \frac{e(l_{\mathrm{sl}}, g_{\mathrm{sl}})}{g_{\mathrm{sl}}} + \frac{e(l_{\mathrm{tr}}, g_{\mathrm{tr}})}{g_{\mathrm{tr}}} \tag{19}$$

where $e(l, g) = |l - g|$ denotes absolute error between value l achieved in regulation and goal value g, and subscripts "th", "sl", and "tr" refer to the terms of loading rate, power output of slack node, and the transmission loss, respectively. Control error reflects accuracy of power line flow regulation.

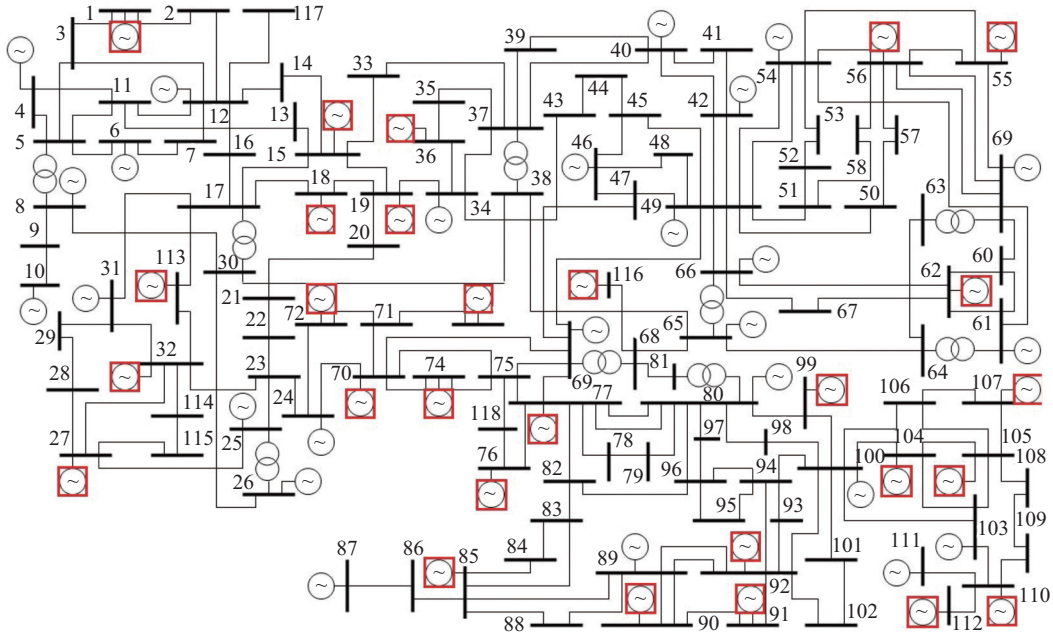

Fig. 4. Node diagram of the IEEE118-bus system in this work.

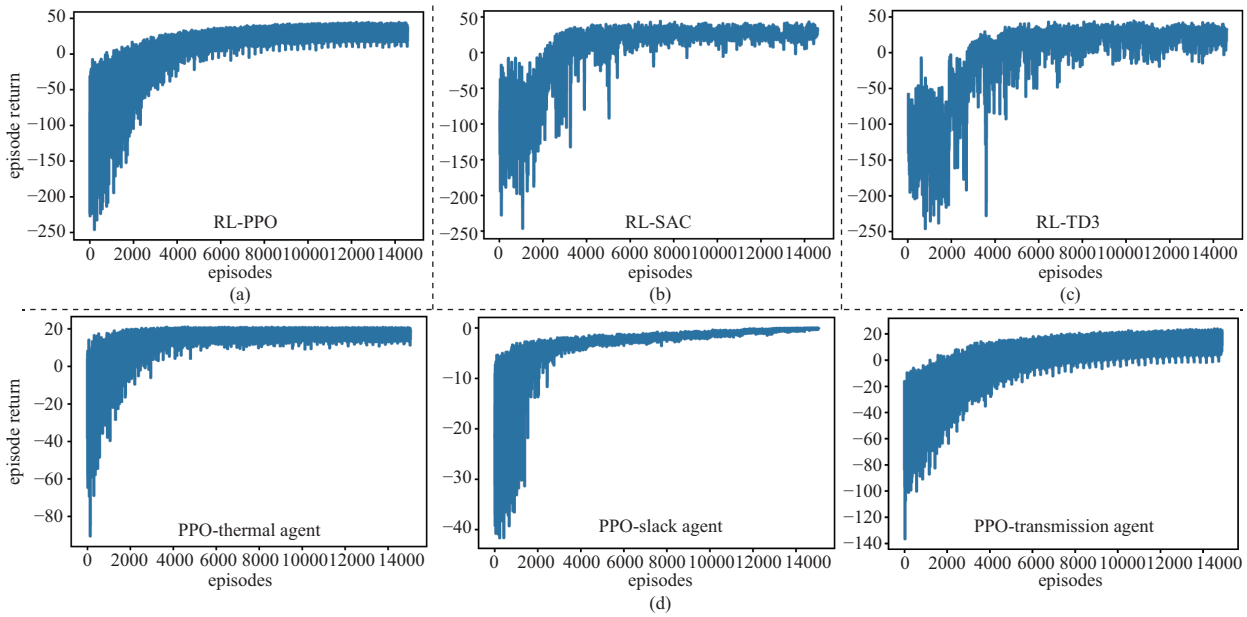

Fig. 5. The training curves in terms of the episode return for (a) RL-PPO, (b) RL-SAC, (c) RL-TD3, and (d) the thermal agent, slack agent and transmission agent in our HTP (cf. Table I).

Figure 5 shows training curves in terms of episode return for the three compared methods and 3 agents used in our approach HTP. We can figure out episode return for all agents gradually increases and converges, which indicates the agents have learned an effective policy.

We demonstrate an example of planning process in Fig. 6. At beginning of the task, higher planner π_{H} outputs sub-task "data prepare" to select an initial power state, and lower planner π_{L} outputs atomic tasks of "select power flow" and "run power flow" to complete "data prepare" sub-task. Next, classic planner π_{c} in π_{H} outputs 3 sub-tasks, i.e., "thermal limit", "slack limit" and "transmission loss". Then, state abstraction module starts to map power state to an abstract power state: 1) max loading percent: 116%; 2) slack node output: 300.5 MW; 3) transmission loss: 88 MW. We build the abstract TP-MDP with current abstract state as initial state. After that, abstract planner π_{a} determines optimal sub-task to be executed and starts lower planner π_{L} to plan on atomic task level. The above steps will be repeated until goal is reached.

Figure 7 shows an illustration of planning result. Upper part in the figure shows how abstract state changes (in the form of percentage) as a series of sub-tasks and atomic tasks are executed in the planning procedure. We can find execution of the three sub-tasks regulating the power grid indeed drive

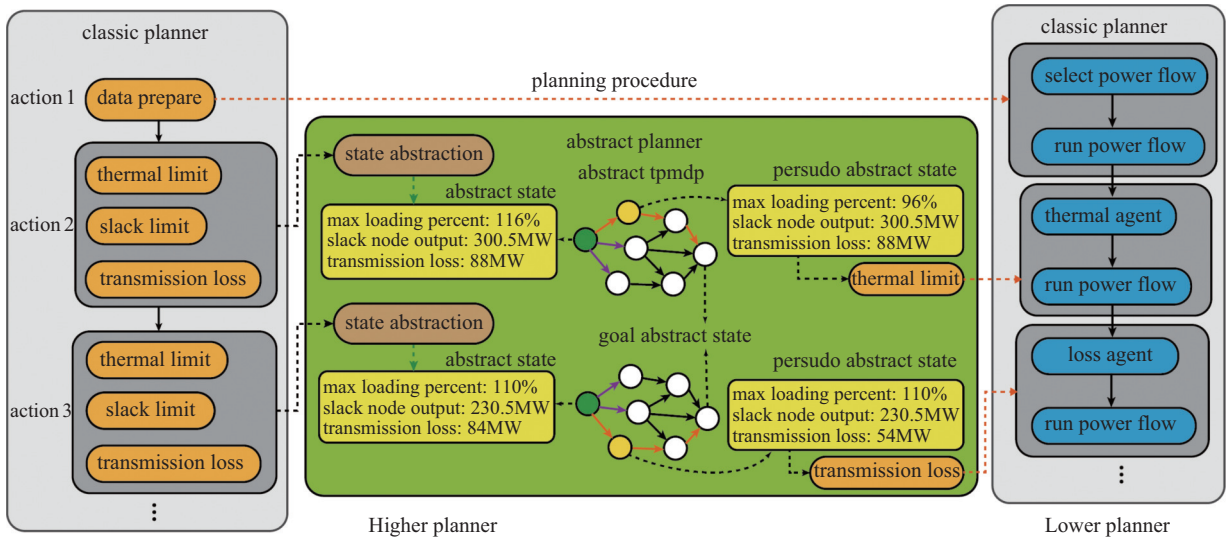

Fig. 6. Example of planning procedure on IEEE118-bus system.

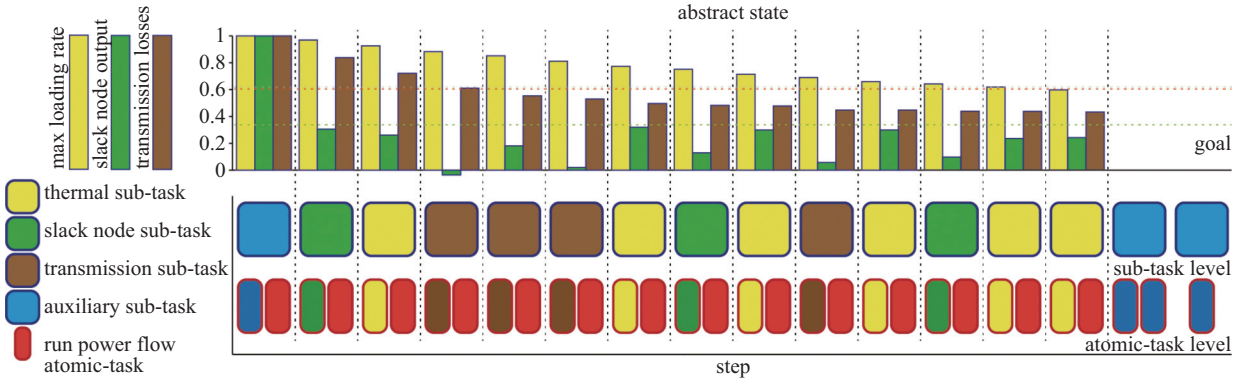

Fig. 7. Illustration of the planning result.

abstract state to the goal of the task. Lower part shows how the sub-tasks are completed by execution of atomic tasks.

In Table II, we compare our approach with baselines on 2,880 test examples and report the mean and standard deviation of control steps taken to reach the goal. Table shows that RL-PPO outperforms the other two DRL compared methods as a whole. Moreover, results clearly demonstrate our HTP approach takes fewer steps on average than the three compared DRL methods, and exhibits a smaller standard deviation. As difficulty of regulation goal increases (i.e., from goal 1 to goal 3), RL-PPO, RL-SAC, and RL-TD3 exhibit a significant increase in number of control steps necessary to reach the goal. In contrast, HTP effectively maintains its efficiency by consistently taking fewer control steps across all goals. Furthermore, we also assess performance of HTPs with initial K_{dp} and updated K_{dp}, and observe incorporation of updated experience during the planning process slightly enhances efficiency of regulation. Specifically, HTP reduces average number of steps by 25.48% compared with RL-PPO when taking initial K_{dp}, and by 26.16% with introduction of updated K_{dp}.

We compare our approach with baselines in terms of control error on 2,880 test examples and report mean and standard deviation in Table III. We notice our HTP approach with

TABLE II
MEAN AND STANDARD DEVIATION (SHOWN IN BRACKETS) OF THE CONTROL STEPS TO REACH THE GOAL ON IEEE 118-BUS SYSTEM

Goal	RL-SAC	RL-TD3	RL-PPO	HTP	
				initial K_{dp}	updated K_{dp}
goal 1	13.59 (8.196)	23.94 (19.773)	14.60 (3.362)	14.26 (1.036)	14.12 (1.156)
goal 2	20.19 (15.087)	41.43 (11.002)	18.82 (12.779)	15.41 (2.569)	15.38 (2.540)
goal 3	35.29 (18.142)	45.65 (3.770)	29.88 (17.700)	17.50 (5.381)	17.22 (5.068)

TABLE III
MEAN AND STANDARD DEVIATION (SHOWN IN BRACKETS) OF THE CONTROL ERRORS TO REACH THE GOAL ON IEEE 118-BUS SYSTEM

Goal	RL-SAC	RL-TD3	RL-PPO	HTP	
				initial K_{dp}	updated K_{dp}
goal 1	0.162 (0.045)	0.113 (0.048)	0.179 (0.037)	0.163 (0.009)	0.164 (0.008)
goal 2	0.118 (0.039)	0.111 (0.059)	0.107 (0.032)	0.091 (0.016)	0.089 (0.015)
goal 3	0.076 (0.032)	0.185 (0.085)	0.074 (0.031)	0.041 (0.017)	0.038 (0.016)

updated K_{dp} achieves smallest control error on average, and reduces average control error across all three goals by 19.17% compared with RL-PPO.

Figure 8 illustrates performance of baselines and our approach on an example, where each column corresponds to a goal, and each row to an objective of the goal, and the dashed line indicates steps the corresponding approach takes to reach the goal. Each colored dashed line denotes the steps that the corresponding approach takes to reach the goal, and the missing lines indicate that the corresponding approach does not reach the goal within the maximal number of steps shown in Fig. 8. We find in each case, HTP takes fewer steps than baseline RL-PPO. In addition, HTP can reduce distance to goal state more rapidly in most cases. We also illustrate the effect of dynamic planning knowledge graph K_{dp} on the task. Compared with HTP with initial K_{dp} (HTP-init K_{dp}), HTP with updated K_{dp} (HTP-updated K_{dp}) can reach the goal in fewer steps.

D. Results of Planning on IEEE 300-bus System

The IEEE 300-bus system consists of 69 generators including 19 renewable units, 304 transmission lines, and 195 loads. We designed a total of 9 patterns regarding loads and renewable units. We totally collected 60,480 samples as a dataset, 57,600 for training and the rest 2,880 for test.

We designate 3 different goals for the task of power line flow regulation as follows. Goal 1—max loading percent: 100%; max slack node output: 250 MW; max transmission losses: 200 MW. Goal 2—max loading percent: 100%; max slack node output: 150 MW; max transmission losses: 180 MW. Goal 3—max loading percent: 100%; max slack node output: 100 MW; max transmission losses: 150 MW. Difficulty increases to achieve successful regulation across goal 1 to goal 3.

We compare our approach with baselines on 2,880 test examples. Table IV illustrates average performance in terms of mean and standard deviation of control steps and control errors across all the three goals. Table IV shows RL-PPO is superior to the other two DRL baselines. Moreover, HTP takes fewer steps than the other two DRL baselines and exhibits a smaller

TABLE IV
MEAN AND STANDARD DEVIATION (SHOWN IN BRACKETS) OF THE CONTROL STEPS AND CONTROL ERRORS TO REACH THE GOAL ON IEEE 300-BUS SYSTEM

	RL-SAC	RL-TD3	RL-PPO	HTP	
				initial K_{dp}	updated K_{dp}
control steps	22.35 (14.452)	28.74 (18.424)	14.28 (11.942)	18.42 (10.311)	13.30 (12.109)
control errors	0.262 (0.168)	0.235 (0.195)	0.205 (0.157)	0.259 (0.383)	0.179 (0.162)

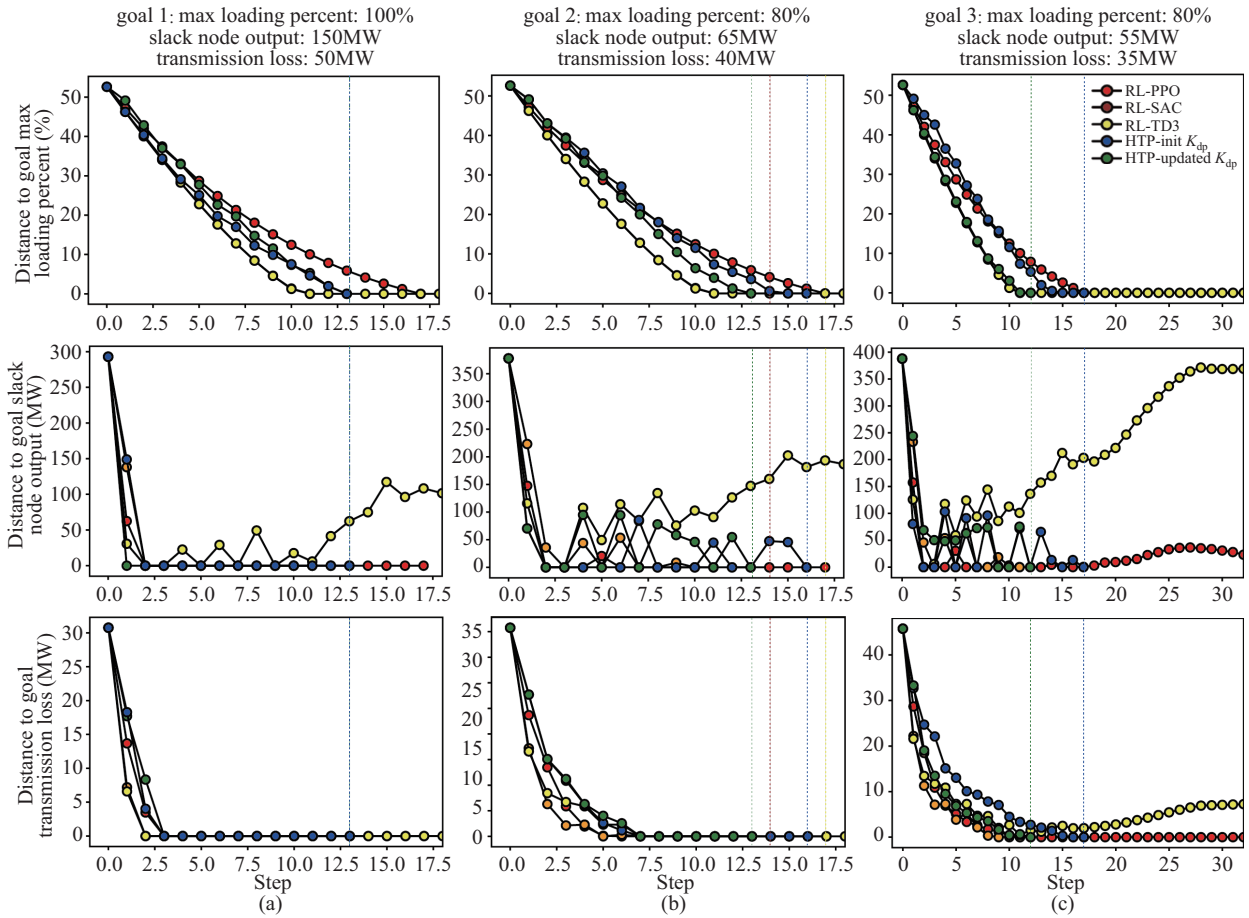

Fig. 8. Performance comparison between our approach and 3 compared approaches for 3 different goals on the IEEE 118-bus system. (a) Distance to goal 1. (b) Distance to goal 2. (c) Distance to goal 3.

standard deviation. Furthermore, we assess performance of HTPs with initial K_{dp} and updated K_{dp}, and observe the latter leads to an improvement in terms of both metrics compared with the former, which indicates incorporation of updated experience during the planning process helps accomplish regulation more effectively and more efficiently. Specifically, HTP with updated K_{dp} reduces average control steps by 6.86% and the average control errors by 12.68% compared with RL-PPO.

V. Conclusion

In this paper, we has proposed a hierarchical task planner (HTP) approach for the task of power line flow regulation. We first introduce a three-level task hierarchy to model the task. In the proposed HTP, we introduce a higher planner and a lower planner working alternately to plan execution of task units on different levels. In addition, we introduce a two-layer knowledge graph to assist planners. Experimental results conducted on the IEEE 118-bus and 300-bus systems demonstrate the proposed HTP outperforms state-of-the-art DRL algorithm. HTP improves average efficiency and control error by 26.16% and 19.17% with default dynamic planning knowledge graph, and by 6.86% and 12.68% with update of knowledge graph on IEEE 118-bus and 300-bus system.

References

[1] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, and S. J. Wright, "Distributed MPC strategies with application to power system automatic generation control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1192–1206, Nov. 2008.

[2] S. J. Russell, *Artificial Intelligence: a Modern Approach*. 3rd ed., Upper Saddle River, NJ: Prentice Hall, 2010.

[3] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, "Mastering Atari, Go, chess and Shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, Dec, 2020.

[4] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv: 1707.06347, 2017.

[5] H. J. Zhang, N. Yang, W. Huangfu, K. P. Long, and V. C. M. Leung, "Power control based on deep reinforcement learning for spectrum sharing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 4209–4219, Jun. 2020.

[6] B. Zhang, X. Lu, R. S. Diao, H. F. Li, T. Lan, D. Shi, and Z. W. Wang, "Real-time autonomous line flow control using proximal policy optimization," in *Proceedings of 2020 IEEE Power & Energy Society General Meeting*, 2020, pp. 1–5.

[7] H. Tang, K. Lv, B. Bak-Jensen, J. R. Pillai, and Z. F. Wang, "Deep neural network-based hierarchical learning method for dispatch control of multi-regional power grid," *Neural Computing and Applications*, vol. 34, no. 7, pp. 5063–5079, Apr. 2022.

[8] Y. Ji, J. H. Wang, J. C. Xu, X. K. Fang, and H. G. Zhang, "Real-time energy management of a microgrid using deep reinforcement learning," *Energies*, vol. 12, no. 12, pp. 2291, Jun. 2019.

[9] Y. F. Zhang, Q. Ai and Z. Y. Li, "Intelligent demand response resource trading using deep reinforcement learning," *CSEE Journal of Power and Energy Systems*, doi: 10.17775/CSEEJPES.2020.05540.

[10] M. Kamel, R. C. Dai, Y. W. Wang, F. X. Li and G. Y. Liu, "Data-driven and model-based hybrid reinforcement learning to reduce stress on power systems branches, " *CSEE Journal of Power and Energy Systems*, vol. 7, no. 3, pp. 433–442, May 2021.

[11] J. Li, S. Chen, X. Y. Wang and T. J. Pu, "Load Shedding Control Strategy in Power Grid Emergency State Based on Deep Reinforcement Learning," *CSEE Journal of Power and Energy Systems*, vol. 8, no. 4, pp. 1175–1182, Jul. 2022.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: an Introduction*, 2nd ed., Cambridge: MIT Press, 2018.

[13] D. M. Lyu, F. K. Yang, B. Liu, and S. Gustafson, "SDRL: interpretable and data-efficient deep reinforcement learning leveraging symbolic planning," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019, pp. 2970–2977.

[14] F. K. Yang, D. M. Lyu, B. Liu, and S. Gustafson, "PEORL: integrating symbolic planning and hierarchical reinforcement learning for robust decision-making," in: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018, pp. 4860–4866.

[15] A. Faust, K. Oslund, O. Ramirez, A. Francis, L. Tapia, M. Fiser, and J. Davidson, "PRM-RL: long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning," in *Proceedings of 2018 IEEE International Conference on Robotics and Automation*, 2018, pp. 5113–5120.

[16] T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker, "A unifying framework for reinforcement learning and planning," Frontiers in Artificial Intelligence, vol. 5, p. 908353, 2022.

[17] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: learning skills without a reward function," in *Proceedings of the 7th International Conference on Learning Representations*, 2018.

[18] A. Sharma, S. X. Gu, S. Levine, V. Kumar, and K. Hausman, "Dynamics-aware unsupervised discovery of skills," in *Proceedings of the 8th International Conference on Learning Representations*, 2020.

[19] X. M. Dong, H. Sun, C. F. Wang, Z. H. Yun, Y. M. Wang, P. H. Zhao, Y. Y. Ding, and Y. Wang, "Power flow analysis considering automatic generation control for multi-area interconnection power networks," *IEEE Transactions on Industry Applications*, vol. 53, no. 6, pp. 5200–5208, Nov./Dec. 2017.

[20] P. Kundur, G. K. Morison, and L. Wang, "Techniques for on-line transient stability assessment and control," in *Proceedings of 2000 IEEE Power Engineering Society Winter Meeting. Conference Proceedings*, 2000, pp. 46–51.

[21] R. S. Sutton and A. G. Barto, "*Introduction to Reinforcement Learning*," Cambridge: MIT Press, 1998.

[22] M. Hanheide, M. Göbelbecker, G. S. Horn, A. Pronobis, K. Sjöö, A. Aydemir, P. Jensfelt, C. Gretton, R. Dearden, M. Janicek, H. Zender, G. J. Kruijff, N. Hawes, and J. L. Wyatt, "Robot task planning and explanation in open and uncertain worlds," *Artificial Intelligence*, vol. 247, pp. 119–150, Jun. 2017.

[23] L. X. Yang, Q. Y. Sun, N. Zhang, and Z. W. Liu, "Optimal energy operation strategy for we-energy of energy internet based on hybrid reinforcement learning with human-in-the-loop," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 1, pp. 32–42, Jan. 2022.

[24] J. J. Duan, D. Shi, R. S. Diao, H. F. Li, Z. W. Wang, B. Zhang, D. S. Bian, and Z. H. Yi, "Deep-reinforcement-learning-based autonomous voltage control for power grid operations," *IEEE Transactions on Power Systems*, vol. 35, no. 1, pp. 814–817, Jan. 2020.

[25] R. Rocchetta, L. Bellani, M. Compare, E. Zio, and E. Patelli, "A reinforcement learning framework for optimal operation and maintenance of power grids," *Applied Energy*, vol. 241, pp. 291–301, May 2019.

[26] Q. H. Huang, R. K. Huang, W. T. Hao, J. Tan, R. Fan, and Z. Y. Huang, "Adaptive power system emergency control using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1171–1182, Mar. 2020.

[27] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, Mar. 2012.

[28] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. P. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm" arXiv: 1712.01815, 2017.

[29] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," arXiv: 1812.05905, 2019.

[30] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 1587–1596.

Chenxi Wang received the B.S. degree in Engineering from Xi'an Jiaotong University, China, in 2018, where he is currently pursuing the Ph.D. degree with the School of Electronic and Information Engineering. His research interests include machine learning and deep reinforcement learning.

Youtian Du (M'09) received the Ph.D. degree from Tsinghua University, in 2008. He is currently a Professor with the Faculty of Electronic and Information Engineering, Xi'an Jiaotong University. His current research interests include cross-media analysis and reasoning, knowledge extracting and representation, and machine learning.

Yanhao Huang received an M.S. degree in Power System Automation, Wuhan University, Wuhan, China, in 2004. He works in Power System Department, China Electric Power Research Institute (CEPRI). His research interests include power system simulation, intelligent technologies and electric power big data.

Yuanlin Chang received the M.S. degree in Computer Science from Xi'an Jiaotong University, China, in 2022, where he is currently pursuing the Ph.D. degree with the School of Electronic and Information Engineering. His research interests include knowledge-based reinforcement learning and deep learning.

Zihao Guo received the B.S. degree in the School of Control and Computer Engineering from North China Electric Power University. He is currently pursuing the M.S. degree with the School of Electronic and Information Engineering from Xi'an Jiaotong University. His research interests include machine learning and reinforcement learning.