

Nested Reinforcement Learning Based Control for Protective Relays in Power Distribution Systems

Dongqi Wu, Xiangtian Zheng, Dileep Kalathil, Le Xie

Abstract—This paper envisions a new control architecture for the protective relay setting in future power distribution systems. With deepening penetration of distributed energy resources at the end users level, it has been recognized as a key engineering challenge to redesign the protective relays in the future distribution system. Conceptually, these protective relays are the discrete ON/OFF control devices at the end of each branch and node in a power network. The key technical difficulty lies in how to set up the relay control logic so that the protection could successfully differentiate heavy load and faulty operating conditions. This paper proposes a new nested reinforcement learning approach to take advantage of the structural properties of distribution networks and develop a new set of training methods for tuning the protective relays.

I. INTRODUCTION

This paper is motivated by the increasing need to re-design the control architecture of protective relays in the power distribution systems. The goal of protective relays is to detect abnormal conditions, such as short circuit and equipment failures, and isolate the corresponding elements to prevent possible cascading destruction. The key design criteria for protective relays in the power distribution system is to properly isolate faults under abnormal conditions while not tripping under normal operating conditions. Since the protective relays are installed at all the nodes and branches, tripping of a protective relay would have consequences beyond the immediate neighboring device in the system. Therefore, the art and science of designing a protective relay system lies in how to trade-off different protective relay tripping during faulty situations. With increasing level of uncertainties in line flow patterns due to distributed energy resources, the design of a intelligent relay system has become the key engineering challenge to fully realize the potential of a truly low-carbon energy system in the future. This paper directly addresses this challenge of how to re-design the protective relay systems in the distribution grid.

This paper focuses on the re-design of the control logic for overcurrent relays. Overcurrent relays are the most widely used protective relays in the power grid. Overcurrent relays use the current magnitude as the indicator of faults. When a short-circuit fault occurs, the fault current is typically much larger than the nominal current under the normal conditions. The operating principle of this kind of relay is to trip the line if the measured current exceeds a pre-fixed threshold. This threshold is usually determined based on a number of

heuristics that account for the topology of the network and feeder capacity.

In the case of possible operation failure of any relay, some coordination between adjacent relays is necessary to avoid catastrophic outcomes. This is typically achieved with a primary relay - backup relay coordination. If a fault occurs in the assigned region of a given relay, it should act as the primary relay and trip. If (and only if) the primary relay fails to trip, the adjacent upstream (towards the feeder) relay should trip. Since there is no explicit communication between the relays, this coordination is achieved implicitly using an ‘inverse time curve’ [1]. The basic idea is to design relays in such a way that higher fault current results in shorter tripping time. Since the primary relay is closer to the fault, the measured current will be higher and it will trip faster. If the primary relay fails, the backup relay will work but only after some time delay indicated by the inverse time curve.

Successful operation of conventional overcurrent relays rely on two crucial assumptions: (i) nominal operation currents are always less than the fault current, (ii) the current measurements are always higher for the relays that are closer to the fault. With the increasing penetration of distributed energy resources, both assumptions are likely to be rendered invalid. This is due to the fact that distributed energy resources such as solar panels and batteries may create reverse power flows from the edge of the grid to the substation.

An efficient control algorithm for relay protection should be able to: (i) reduce the operation failures as low as possible, (ii) identify the fault as soon as possible, and (iii) adapt robustly against the changes in the operating conditions, like shift in the load profile. A unified approach that can exploit the availability of huge amounts of real-time sensor data from the power distribution systems, recent advances in machine learning, along with domain knowledge of the power systems operations is necessary to achieve these objective, especially in the context of next generation power systems.

Related work: Most studies of improving the performance of the over-current relays focus on the aspects of coordination [2] [3], fault detection [4] and fault section estimation [5]. Among various possible methods, machine learning is popular for advanced over-current relays. Neural networks [6] [7] are applied to determine the coefficients of the inverse-time over-current curve. Other research work based on support vector machine [8] [9] directly determine the operation of relays. However, most of these learning techniques do not explicitly explore the dynamic nature of the protective relay setting. As the power network grows in its complexity and flow patterns, it is often difficult to

The authors are with the Department of Electrical and Computer Engineering, Texas A&M University, Texas, United States. Email: {dqwu, zxt0515, dileep.kalathil, le.xie}@tamu.edu

differentiate normal setting from a faulty one simply from a snapshot of measurements.

Reinforcement learning (RL) is a class of machine learning that focuses on *learning to control* unknown dynamical systems. Unlike the other two classes of machine learning, supervised learning and unsupervised learning, which typically focus on static systems, RL methodology explicitly includes the tools to characterize the dynamical nature of the system that it tries to learn. In RL, a learning agent sequentially interacts with a (dynamical) system by observing the states of the system and by selecting appropriate control actions. The system state evolves stochastically depending on the control action of the agent and according to the (unknown) model of the system. After each interaction, the agent receives a reward. The goal of the agent is to learn a control policy, which specifies the optimal action to take given the state of the system, in order to maximize the cumulative reward.

Last few years have seen significant progresses in deep neural networks based RL approaches for controlling unknown dynamical systems, with applications in many areas like playing games [11], locomotion [12] and robotic hand manipulation [13]. This has also led to addressing many power systems problems using the tools from RL, as detailed in the survey [14]. RL is indeed the most appropriate machine learning approach for a large class of power systems problems because of the inherent stochastic and dynamical nature of the power systems. RL based applications in power systems include electricity markets [15] [16], voltage control [17], automatic generation control [18], demand control [19], and angle stability control [20] [21]. However, little effort has been made for using RL for relay protection control. The closest work [22] discusses about using a centralized Q-learning algorithm to determine the protection strategy for a relay network with full communication between them. The prerequisite of global communication leaves this method impractical.

Our contributions: We propose a novel *nested reinforcement learning* algorithm for optimal relay protection control for a network of relays in a power distribution network. We don't assume any explicit communication between the relays. We formulate the relay protection control as a multi-agent RL problem where each relay acts as an agent, observes only its local measurements and takes control actions based on this observation. Multi-agent RL problems are known to be intractable in general and convergence results are sparse. We overcome this difficulty by cleverly exploiting the underlying radial structure of power distribution systems. We argue that this structure imposes only a one directional influence pattern among the agents, starting from the end of the line to the feeder. Using this structure, we develop a nested training procedure for the network of relays. Unlike generic multi-agent RL algorithms which often exhibit oscillations and even non-convergence in training, our nested RL algorithm converges fast in simulations. The converged policy far outperforms the conventional threshold based relay protection strategy in terms of failure rates, robustness to change in the operation conditions, and speed in responses.

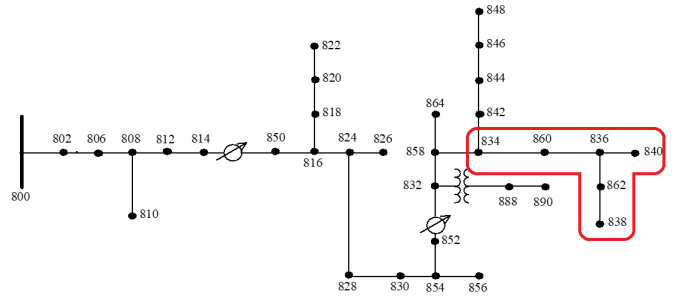


Fig. 1: IEEE34 node test feeder

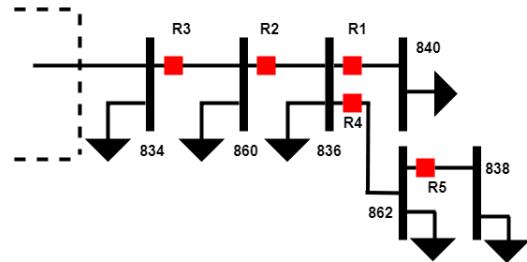


Fig. 2: Protective relays in a radial network

This paper is organized as follows. Section II formulates the relay operation problem. Section III gives a brief review on RL. Section IV provides our new algorithm. Section V presents simulation studies that show the efficiency of the proposed method. Concluding remarks are presented in Section VI.

II. PROBLEM FORMULATION

In order to precisely characterize the operation of protective relays, we first explain what ideal relays are supposed to do using a concrete setting given in Fig. 2. This is a small section of the larger standard IEEE 34 node test feeder [23] shown in Fig. 1. There are five relays protecting five segments of the distribution line.

Desirable operation of the relays is as follows. Each relay is located to the right of a bus (node). Each relay needs to protect its own region, which is between its own bus and the first down-stream bus. Relays are also required to provide backup for its first downstream neighbor: when its neighbor fails to operate, it needs to trip the line and clears the fault. For example, in Fig. 2, if a fault occurs between bus 862 and 838, relay 5 is the main relay protecting this segment and it should trip the line immediately. If relay 5 fails to work, relay 4, which provides backup for relay 5, needs to trip the line instead. The time delay between detecting and clearing a fault should be as short as possible for primary relays, while backup relays should react slower to ensure that they are only triggered when the corresponding main relay is not working.

According to the above description, the desired operation of a network of protective relays can be formalized as follows. Suppose there are n relays. Denote the control action of i th relay as a_i and the index of its downstream neighbor as n_i . For a_i , 1 means to trip while 0 means to hold. Let X_{p_i}

and X_{b_i} respectively be the primary and backup protection region of relay i . Suppose x_f is the location of the fault, then the ideal control action a_i of relay i is

$$a_i = \mathbb{1}((x_f \in X_{P_i}) \cup ((x_f \in X_{b_i}) \cap (a_{n_i} = 0))) \quad (1)$$

where $\mathbb{1}(\cdot)$ is an indicator function.

However, in practice each relay knows only the local measurements like voltage and current. In particular, relay i is not aware of downstream neighbors' actions a_{n_i} and the exact location of the fault x_f . So, each relay i needs a local control policy π_i that maps the local observation s_i to control action a_i , i.e., $a_i = \pi_i(s_i)$. These local control policies are to be designed in such a way to enable an implicit coordination between the relays in the network to achieve the a global protection strategy. In following, we propose a multi-agent reinforcement learning approach for addressing this problem.

III. MARKOV DECISION PROCESSES AND REINFORCEMENT LEARNING

Before formulating the relay protection problem using the RL approach, we first give a brief review of some basic terminologies in RL.

Markov decision processes (MDP) is a canonical formalism for stochastic control problems. The goal is to solve sequential decision making (control) problems in stochastic environments where the control actions can influence the evolution of the state of the system. An MDP is modeled as tuple $(\mathcal{S}, \mathcal{A}, R, P, \gamma)$ where \mathcal{S} is the state space, \mathcal{A} is the action space. $P = (P(\cdot|s, a), (s, a) \in \mathcal{S} \times \mathcal{A})$ are the state transition probabilities. $P(s'|s, a)$ specifies the probability of transition to s' upon taking action a in state s . $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, and $\gamma \in [0, 1]$ is the discount factor.

A policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ specifies the control action to take in each possible state. The performance of a policy is measured using the metric *value of a policy*, V_π , defined as

$$V_\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s], \quad (2)$$

where $R_t = R(s_t, a_t)$, $a_t = \pi(s_t)$, $s_{t+1} \sim P(\cdot|s_t, a_t)$. The optimal value function V^* is defined as $V^*(s) = \max_{\pi} V_\pi(s)$. Given V^* , the optimal policy π^* can be calculated using the Bellman equation as

$$\pi^*(s) = \arg \max_{a \in \mathcal{A}} (R(s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, a) V^*(s')). \quad (3)$$

Similar to the value function, Q-value function of a policy π , Q_π , is defined as

$$Q_\pi(s, a) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s, a_0 = a] \quad (4)$$

Optimal Q-value function Q^* is also defined similarly, $Q^*(s, a) = \max_{\pi} Q_\pi(s, a)$. Optimal Q-value function will help us to compute the optimal policy directly without using the Bellman equation, as $\pi^*(s) = \arg \max_{a \in \mathcal{A}} Q^*(s, a)$

Given an MDP formulation, the optimal value/Q-value function (V^*/Q^*) or the optimal control policy (π^*) can be

computed using dynamic programming methods like value iteration or policy iteration [24]. However, these dynamic programming method requires the knowledge of the full model of the system, namely, the transition probability P and reward function R . In most real world applications, the stochastic system model is either unknown or extremely difficult to model. In the protective relay problem, the transition probability represents all the possible stochastic variations in voltage and current in the network, due to a large number of scenarios like weather (and the resulting shift in demand/supply) and renewable energy generation. In such scenarios, the optimal policy has to be *learned* from sequential state/reward observations.

Reinforcement learning is a method for computing the optimal policy for an MDP when the model is unknown. RL achieves this without explicitly constructing an empirical model. Instead, it directly learns the optimal Q-value function or optimal policy from the sequential observation of states and rewards.

Q-learning is one of the most popular RL algorithms which learn the optimal Q^* from the sequence of observations (s_t, a_t, R_t, s_{t+1}) . Q-learning algorithm is implemented as follows. At each time step t , the RL agent updates the Q-function Q_t as

$$Q_{t+1}(s_t, a_t) = (1 - \alpha_t) Q_t(s_t, a_t) + \alpha_t (R_t + \gamma \max_b Q_t(s_{t+1}, b)) \quad (5)$$

where α_t is the step size (learning rate). It is known that if each-state action pairs is sampled infinitely often and under some suitable conditions on the step size, Q_t will converge to the optimal Q-function Q^* [24].

Using a standard tabular Q-learning algorithm as described above is infeasible in problems with continuous state/action space. To address this problem, Q-function is typically approximated using a deep neural network, i.e., $Q(s, a) \approx Q_w(s, a)$ where w is the parameter of the neural network. Deep neural networks can approximate arbitrary functions without explicitly designing the features and this has enabled tremendous success in both supervised learning (image recognition, speech processing) and reinforcement learning (AlphaGo games) tasks.

In Q-learning with neural network based approximation, the parameters of the neural network can be updated using stochastic gradient descent with step size α as

$$w = w + \alpha \nabla Q_w(s_t, a_t) (R_t + \gamma \max_b Q_w(s_{t+1}, b) - Q_w(s_t, a_t)) \quad (6)$$

Unlike supervised learning algorithm, the data samples (s_t, a_t, R_t, s_{t+1}) obtained by an RL algorithm is correlated in time due to the underlying system dynamics. This often leads to a very slow convergence or non-convergence of the gradient descent algorithms like (6). The idea of *experience replay* is to break this temporal correlation by randomly sampling some data points from a buffer of previously observed (experienced) data points to perform the gradient

TABLE I: Relay State Space

State Variable	Description
Voltage	Voltage measurements of past m timesteps
Current	Current measurements of past m timesteps
Status	Current breaker state: open or close
Counter	Current value of the time counter

TABLE II: Relay Action Space

Action	Description
Countdown	Continue the counter
Set	Set counter to value 1 - 9
Reset	Stop activated counter

step in (6). New observations are then added to the replay buffer and the process is repeated.

In the gradient descent equation (6), the target $R_t + \gamma \max_b Q_w(s_{t+1}, b)$ depends on the neural network parameter w , unlike the targets used for supervised learning which are fixed before learning begins. This often leads to poor convergence in RL algorithms. The idea of *target network* is used to address this issue. A separate (target) neural network is used for maintaining the target value for gradient descent. The target network is kept fixed for multiple steps but updated periodically.

The combination of neural networks, experience replay and target network forms the core of the DQN algorithm [25]. In the following, we will use DQN as one of the basic block for our nested RL algorithm.

IV. NESTED REINFORCEMENT LEARNING FOR CONTROL OF PROTECTIVE RELAYS

We model the protective relays as collection of RL agents. Each relay can only observe its local measurements of voltage and current. Each relay also knows the status of the local current breaker circuits, i.e., if it is open or closed. Since relays don't observe the measurements at other relays, an implicit coordination mechanism is also needed in each relay. This is achieved by including a local counter that ensures the necessary time delay in its operation as backup relay. These variables constitute the state $s_i(t)$ of each relay i at time t . Table I summarizes this state space representation. Note that the state includes the past m measurements.

To define the action space, we first specify the possible actions each relay can take. When a relay detects a fault it will decide to trip. However, to facilitate the coordination between the network of relays, rather than tripping instantaneously, it will trigger a counter with a time countdown, indicating the relay will trip after certain time steps. If the fault is cleared by another relay during the countdown, the relay will reset the counter to prevent mis-operation. Table II summarizes the action space of each relay. So, the action of relay i at time t , $a_{i,t}$, is one of these 11 possible values.

The reward given to each relay is determined by its current action and fault status. A positive reward occurs if, i) it remains closed during normal conditions, ii) it correctly operates after a fault in its assigned region or in its first

TABLE III: Rewards

Condition	Trip	Hold
Normal	-150	+3
After fault in main region	+120	-3 for each post-fault step
After fault as backup	+100	-2 for each post-fault step
After fault outside assigned region	-150	+5

downstream region when the corresponding primary relay fails. A negative reward is caused by, i) tripping when there is no fault; 2) tripping after a fault outside its assigned region. The magnitude of the rewards are designed in such a way to facilitate the learning, implicitly signifying relative importance of false positives and false negatives. The reward function for each relay is shown in Table III.

Consider a network with n relays. Define the global state of the network at time t as $\bar{s}_t = (s_{1,t}, s_{2,t}, \dots, s_{n,t})$ and the global action at time t as $\bar{a}_t = (a_{1,t}, a_{2,t}, \dots, a_{n,t})$. Let $R_{i,t}$ be the reward obtained by relay i at time t . It is clear from the description of the system that $R_{i,t}$ depends on the global state \bar{s}_t and global action \bar{a}_t rather than the local state $s_{i,t}$ and local action $a_{i,t}$ of relay i . Define the global reward \bar{R}_t as $\bar{R}_t = \sum_{i=1}^n R_{i,t}$. Note that the (global) state evolution of the network can longer be described by looking at the local transition probabilities because the control actions of the relays affect each others' states. The global dynamics is represented by the transition probability $\bar{P} = \bar{P}(\bar{s}_{t+1}|\bar{s}_t, \bar{a}_t)$.

We formulate the optimal relay protection problem in a network as multi-agent RL problem. The goal is to achieve a global objective, maximizing the cumulative reward obtained by all relays, using only local control laws π_i which maps the local observations $s_{i,t}$ to local control action $a_{i,t}$. Formally,

$$\max_{(\pi_i)_{i=1}^n} \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \bar{R}_t], \quad a_{i,t} = \pi_i(s_{i,t}). \quad (7)$$

Since the model is unknown and there is no communication between relays, each relay has to learn its own local control policy π_i using an RL algorithm to solve (7).

Classical RL algorithms and their deep RL versions typically address only the single agent learning problem. A multi-agent learning environment violates one of the fundamental assumption needed for the convergence of RL algorithms, namely, the stationarity of the operating environment. In a single agent system, for any fixed policy of the learning agent, the probability distribution of the observed states can be described using a stationary Markov chain. RL algorithms are designed to learn only in such a stationary Markovian environment. Multiple agents taking actions simultaneously violate this assumption. Even if the policy of a given agent is fixed, state observations for that agent are no longer according to a stationary Markov chain, as they are controlled by the actions of other agents. Moreover, in our setting, each relay observes only its local measurements which further complicates the problem. A formulation that involves these

two constraints is known as Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [28]. There are existing literatures [29] addressing this kind of problems, but the performance of most algorithms are unstable and the convergence is rarely guaranteed.

We propose an approach to overcome this difficulty of multi-agent RL problem by exploiting the radial structure of power distribution systems. Using this structural insight, we develop a nested RL algorithm to extend the single agent RL algorithm to the multi-agent setting we address.

We use the following training procedure. We start from the very end of the radial network in Fig. 2. The relay protecting the last segment is relay 5, which has no downstream neighbors and can be trained using the single-agent training algorithm described in the previous section. Once the training of relay 5 is complete, it will react to the system dynamics using its learned policy. Since relay 5 only needs to clear local faults (i.e. faults between bus 862 - 838) and ignores disturbances at any other location, its policy will not change according to the change in the policy of other relays. This enables us to train relay 4 with relay 5 operating with a fixed policy (which it learned via the single agent RL algorithm). Since the policy of relay 5 remains fixed when training relay 4, the environment from the perspective of relay 4 remains more or less stationary (except for the possible disturbances due to difference in the local measurements). Similarly, after the training of relay 4 and 1 is complete, relay 2 can be trained with the policy of relay 1, 4 and 5 fixed. This process can be repeated for all the relays upstream to the feeder.

This nested training approach which exploits the nested structure of the underlying physical system allows us to overcome the non stationarity issues presented in generic multi-agent RL settings. Our nested RL algorithm is formally presented in Algorithm 1.

V. SIMULATION RESULTS

In this section we evaluate the performance of our RL algorithm for protective relays. We compare its performance with the conventional threshold based relay protection strategy. We compare the performance in the following metrics: *Failure rate*: We evaluate the percentage of operation failures of relays in four different scenarios: when there is a: (i) fault in the local region, (ii) fault in the immediate downstream region, (iii) fault in a remote region, (iv) no fault in the network.

Robustness to changes in the operating conditions: Relays are trained for a given operating condition, like a specific load profile. We evaluate protective relay strategies when there are changes in such operating conditions.

Response time: Relay protection control is supposed to work immediately after a fault occurs. We evaluate the time taken between the occurrence of a fault and activation of the protection control.

A. Simulation Environment Implementation

We choose the network structure shown in Fig. 1 for simulations. In particular, we focus on the relays in the

Algorithm 1 Nested RL for Radial Relay Network

```

Sort  $\{i | 1 \leq i \leq n\}$  into a vector  $N$  by the ordering of
training
Initialize replay buffer of each relay  $N_i, 1 \leq i \leq n$ 
Initialize Q-value function of each relay  $i$  with random
network weights  $w_i$ 
for relay  $i = 1$  to  $n$  do
  for episode  $k = 1$  to  $M$  do
    Initialize PSS/E simulation environment with ran-
    domly generated system parameters
    for time step  $t = 1$  to  $T$  do
      Observe the state  $s_{i,t}$  of each relay  $N_i$ 
      for relay  $j = 1$  to  $i$  do
        With probability  $\epsilon$  select a random action  $a_{j,t}$ ,
        otherwise select a greedy action
         $a_{j,t} = \arg \max_a Q_{w_j}(s_{j,t}, a)$ 
        Observe the reward  $R_{j,t}$  and next state  $s_{j,t+1}$ 
        Store  $(s_{j,t}, a_{j,t}, R_{j,t}, s_{j,t+1})$  in the replay buffer
        of relay  $N_j$ 
        Sample a minibatch from replay buffer and
        update  $w_j$ 
      end for
      for relay  $j = i + 1$  to  $n$  do
        Select the null action,  $a_{j,t} = 0$ 
      end for
    end for
  end for
end for

```

section of the network shown in Fig. 2. We first describe the simulation environment created for training and testing different relay protection strategies. We implemented the environment using Power System Simulator for Engineering (PSS/E) by Siemens, a commercial power system simulation software. The simulation process is controlled by Python using the official PSSPY interface library and the dynamic simulation module. The environment is wrapped according to the OpenAI Gym [26] format that provides APIs for agents to start, step through and end an episode.

An episode is defined as a short simulation segment that contains a fault. In the beginning of each episode, a random initial operating condition (e.g. generator output, load size) is generated to mimic the load variation in distribution systems. During an episode, a fault is added to the system at a random time-step. The fault is set to have random fault impedance within a range determined according to the proposed model in [27] and occurs at a random location. In order to mimic the real world scenario that necessitates the backup relays, a small probability to ignore a tripping action is added for each relay. This corresponds to the case when the breaker fails as a relay tries to trip the line. In this case, the first upstream relay needs to trip the line instead.

In practice, the load level of distribution systems varies a lot with the time of day. To simulate this, the system is assigned a random trend factor from 70% to 130% in the

TABLE IV: DQN Agent Hyperparameters

Hyperparameter	Value
Hidden Layers	2
Layer Size	128/64
Activation	ReLU/ReLU/Linear
Replay Buffer	10000
Replay Batch	32
Target Soft Update Rate	0.005
Optimizer	Adam, Learning Rate 0.0005
Double DQN	On
Loss	MSE
Discount	0.95

beginning of each episode. In addition to this, each load in the system has its own load multiplier between 80% and 120% of the system-wide trend factor. The capacity of each load is determined by multiplying the base case capacity and the local multiplier. A powerflow solution is then calculated using this random load profile as the initial condition for the dynamic simulation.

B. RL Algorithm Implementation and Training

The RL algorithm is implemented using the open-source library Keras-RL [30]. Since the Python interface of PSS/E used for the environment simulations and the Keras-RL used for training the RL agents are not compatible, they must run on two separate processes simultaneously. In order to enable this, the environment is wrapped as a TCP/IP server that accepts connections through a pre-allocated port and keeps running in the background. RL agents communicate with the server using the same port via *localhost*. Also, to accommodate the TCP/IP requirement, a dedicated encoder/decoder pair is also written to pack the data as byte strings before transmitting through the port. We implemented Algorithm 1 using this setup. Hyperparameters are selected via random search. The final configuration and hyperparameters of the RL algorithm are specified in Table IV. All agents use the same hyperparameters in training.

The learning curves are shown in Fig. 3. In particular, Fig. 3a shows the convergence of episodic reward (cumulative reward obtained in an episode) for relay 5. The thick line indicates the mean value of episodic reward obtained during a trial consisting of 20 independent runs of training. The shaded envelope is bounded by the mean reward \pm standard deviation recorded at the same progress during the trial. Note that the episode reward converges in less than 250 episodes. One episode takes roughly 3 seconds. So, the training converges in less than 750 seconds.

Similarly, Fig. 3b shows the convergence of the false operations for relay 5. In the beginning of training, the false operation rate is really high but it soon converges to a value approximately zero. Fig. 3c shows the learning curve corresponding to the episodic reward of relay 4. The behavior is similar to that of relay 5 though it takes more number of episodes to converge. This is expected because relay 4 has to act both as primary relay and as backup for relay 5, while relay 5 only has to act as the primary relay (c.f. Fig. 2).

So, the control policy of relay 4 is more complex than the policy of relay 5, and hence it takes a longer training time to converge. Also, the standard deviation of episodic reward of relay 4 is slightly higher than relay 5. This is because the reward relay 4 can get in backup relay mode can be smaller than in primary relay mode even for perfect operation.

We omit the learning curves for other relays as they are very similar to that of relay 5 and relay 4.

C. Conventional Relay Protection Strategy

Conventional relay protection strategies are based on a fixed threshold rule. The relay trips if and only if the measured current is greater than a fixed threshold. The optimal threshold is typically computed using a variety of heuristic methods [1]. Since these methods depend on the network parameters like topology, feeder capacity and load size, they may not give the optimal threshold that maximizes the success rate in our setting. So, for a fair comparison with a more powerful RL based algorithm, we compute the threshold that guarantees the optimal performance using a simple statistical approach.

We compute the empirical probability distribution (pdf) of the current measurements before and after the fault from 500 episodes. For example, the pdf of the pre-fault and post-fault current at bus 862 is plotted in Fig. 3d. It is clear from the figure that the distributions of the pre-fault and post-fault currents overlap with each other. This is expected, especially in the power distribution systems, where the load profile varies greatly with the time of day. In fact, when the system is lightly loaded, the fault current with a relatively large fault impedance can be smaller than the normal operating current when the system is heavily loaded. We put a higher weight on faulty scenarios to overcome the imbalanced prior probabilities. The optimal threshold that will maximize the success rate can then be approximated as the crossing point of these two pdfs [31]. This point is marked as the ‘pickup current’ in the figure, and is used as the threshold for the conventional relay protection strategy.

D. Performance Evaluation

In this section we compare the performance of the RL based relay protection strategy and threshold based conventional relay protection strategy. As mentioned above, we focus on three metrics of performance, namely, failure rate, robustness, and response time.

Failure rate: A *false operation* of a relay is the one operation where that relay fails to operate as it supposed to do. There are two kinds of false operations, false-negative and false-positive. A false-positive happens if: (i) relay trips when there is no fault, (ii) relay trips even if the location of fault is outside of the relay’s assigned region, (iii) backup relay trips before the primary relay. A false-negative happens if: (i) relay fails to trip even if the location of the fault is inside its assigned region, (ii) backup relay fails to trip even after its immediate downstream relay has failed.

For the RL based algorithm, we use the parameters obtained after the training. For the conventional relay strategy,

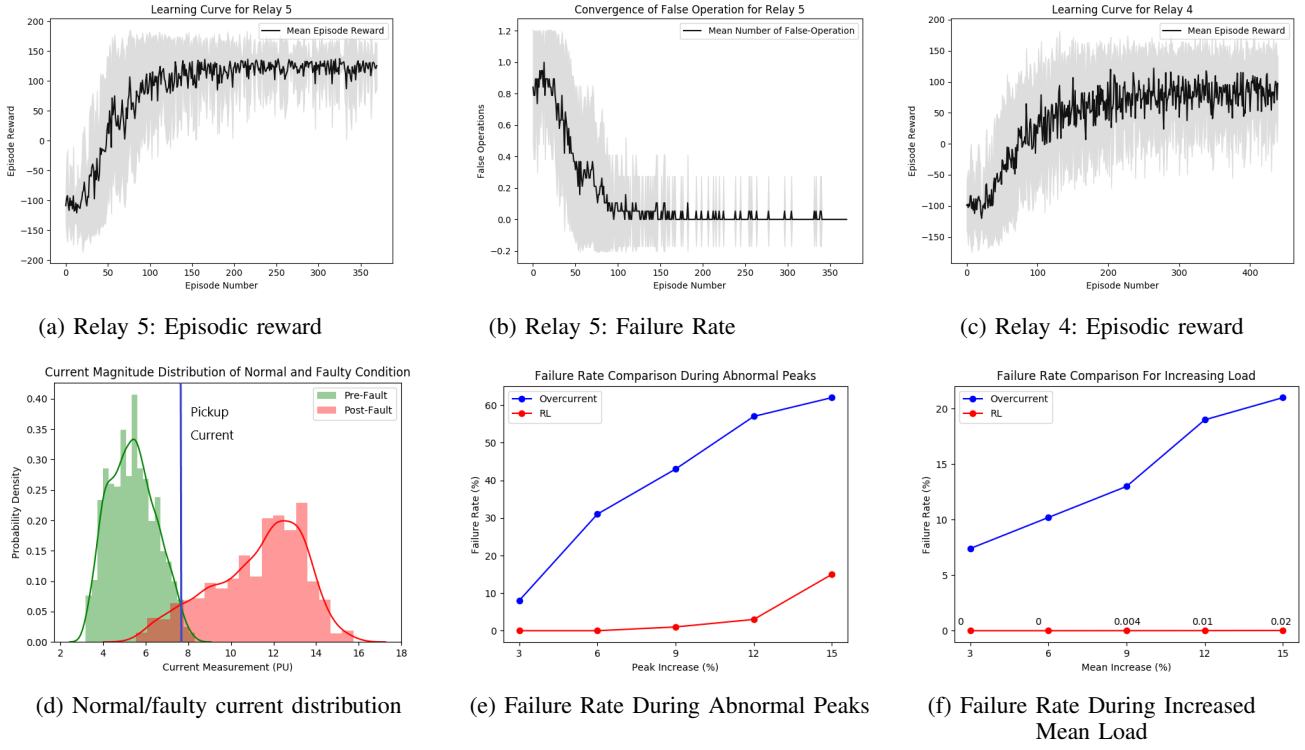


Fig. 3: Convergence Plots of Agents and Comparison of Robustness

we use the optimal threshold computed as described before. We test the performance in four scenarios. Each scenario is tested with 5000 episodes. Failure rate is calculated as the ratio of the number of episodes with failed operations to the total number of episodes. Failure rate comparison is given in Table V. Note that our RL based algorithm remarkably outperforms the conventional relay strategy. For example, in the local fault scenario, the conventional strategy has a failure rate of 7.7% whereas our RL algorithm has a mere 0.26%. Also note that in two scenarios, backup and no fault, even after 5000 random episodes, RL based strategy didn't cause any operation failure. So, we put the failure rate as zero.

TABLE V: False Operation Rate Comparison

Scenario	Expected Operation	Failure Rate	
		Conventional	RL-based
Local Fault	Trip	7.7%	0.26%
Backup	Trip	9.6%	0%
Remote Fault	Hold	3.8%	0.08%
No Fault	Hold	1.8%	0%

Robustness: Load profiles in a distribution system is affected by many events like weather, social activities, renewable generation, and electric vehicles charging schedules. These events can generally cause the peak load to fluctuate and possibly exceed the expected range in the planning stage. Moreover, the electricity consumption is expected to slowly increase each year, reflecting the continuing economy and population growth [32]. This can cause a shift in the mean (and variance) of the load profile. Relay protection control

should be robust to such changes as continually reprogramming relays after deployment is costly.

We first evaluate the robustness in the case of peak load variations. For the clarity of illustration, we focus on relay 5. We vary the peak load up to 15% more than the maximum load used during training. Since we are considering the robustness w.r.t. to the peak load variations, the load capacity used in this test is sampled only from peak load under consideration. For example, the data collected for 3% increase are sampled by setting the system load size between 100% and 103% of the peak load at training. We then test the performance of both relay strategies using the same parameter from the original training, i.e., we don't update the policy to accommodate the change in this load profile.

The performance is shown in Fig. 3e. It can be seen that conventional relay strategy completely fails against such a change in the operating environment as it fails in more than 40% of such scenarios after a 9% increase in the peak load. On the other hand, RL algorithm is remarkably robust at this point with failures in less than 2% scenarios. RL algorithm performance starts to decay noticeably only after 15% increase in the peak load.

We also evaluate the robustness against increase in the mean load. Procedure is same as before and the performance is shown in Fig. 3f. RL algorithm is remarkably robust even after a 15% increase in the mean load. Conventional relay strategy fails completely in this scenario also.

Response time: RL algorithm also shows a very fast tripping speed during the testing. We observed a tripping time of 0.005 second for the primary relay and 0.009 second for the

backup relay. Conventional overcurrent relays uses the time-inverse curves as the ones defined in IEEE C37.112-2018 [33] to determine the time delay for all situations, which gives unnecessary delay even for operations as primary relays. Depending on the curve selected, the minimum delay is at least at the order of 0.1 second.

VI. CONCLUDING REMARKS

This paper proposes a multi-agent reinforcement learning based approach for redesigning the control architecture of protective relay in power distribution systems. We propose a novel nested reinforcement learning algorithm that exploits the underlying physical structure of the network in order to overcome the difficulties associated with standard multi-agent problems. Unlike the generic multi-agent RL algorithms which often fail to converge, our nested RL algorithm converges fast in simulations. The converged policy far outperforms the conventional threshold based relay protection strategy in terms of failure rates, robustness to change in the operation conditions, and speed in responses.

In the future work, we plan to develop analytical guarantees for the convergence of our nested RL algorithm. Future work will also investigate the scalability of the proposed work in much larger system and the impact of network topology on the performance.

REFERENCES

- [1] J. M. Gers and E. J. Holmes, *Protection of Electricity Distribution Networks*, The Institution of Engineering and Technology, 3rd edition, 2011.
- [2] W. El-Khattam and T. S. Sidhu, Restoration of directional overcurrent relay coordination in distributed generation systems utilizing fault current limiter, *IEEE Transactions on Power Delivery*, 2008, vol. 23, no. 2, pp. 576-585
- [3] H. Zhan, C. Wang, Y. Wang, X. Yang, X. Zhang, C. Wu, and Y. Chen, Relay protection coordination integrated optimal placement and sizing of distributed generation sources in distribution networks, *IEEE Transactions on Smart Grid*, 2016, vol. 7, no. 1, pp. 55-65
- [4] P. Dash, S. Samantaray, and G. Panda, "Fault classification and section identification of an advanced series-compensated transmission line using support vector machine", *IEEE transactions on power delivery*, 2007, vol. 22, no. 1, pp. 67-73
- [5] H.-T. Yang, W.-Y. Chang, and C.-L. Huang, "A new neural networks approach to on-line fault section estimation using information of protective relays and circuit breakers", *IEEE Transactions on Power delivery*, 1994, vol. 9, no. 1, pp. 220-230
- [6] P. Mahat, Z. Chen, B. Bak-Jensen and C.L. Bak, "A Simple Adaptive Overcurrent Protection of Distribution Systems With Distributed Generation", *IEEE Transactions on Smart Grid*, 2011, vol.2, no.3, pp 428-437
- [7] H. A. Abyane, K. Faez and H. K. Karegar, "A new method for overcurrent relay (O/C) using neural network and fuzzy logic", TENCON '97 Brisbane - Australia. Proceedings of IEEE TENCON '97. *IEEE Region 10 Annual Conference: Speech and Image Technologies for Computing and Telecommunications (Cat. No.97CH36162)*, Brisbane, Queensland, Australia, 1997, pp. 407-410 vol.1
- [8] D. N. Vishwakarma and Z. Moravej, "ANN based directional overcurrent relay", *2001 IEEE/PES Transmission and Distribution Conference and Exposition. Developing New Perspectives (Cat. No.01CH37294)*, 2001, Atlanta, GA, USA, pp. 59-64 vol.1
- [9] Y. Zhang, M. D. Ilic and O. Tonguz, "Application of Support Vector Machine Classification to Enhanced Protection Relay Logic in Electric Power Grids", *2007 Large Engineering Systems Conference on Power Engineering*, 2007, Montreal, Que., pp. 31-38
- [10] X. Zheng, X. Geng, L. Xie, D. Duan, L. Yang and S. Cui, "A SVM-based setting of protection relays in distribution systems", *2018 IEEE Texas Power and Energy Conference (TPEC)*, 2018, College Station, TX, pp. 1-6.
- [11] D. Silver et al., "Mastering the game of Go with deep neural networks and tree search", *Nature*, 2016, vol. 529, no. 7587, pp. 484
- [12] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning", *arXiv preprint*, 2015, arXiv:1509.02971
- [13] S. Levine, C. Finn, T. Darrell and P. Abbeel, "End-to-end training of deep visuomotor policies", *The Journal of Machine Learning Research*, 2016, vol. 17, no. 1, pp. 1334-1363
- [14] M. Glavic, R. Fonteneau and D. Ernst, "Reinforcement Learning for Electric Power System Decision and Control: Past Considerations and Perspectives", *IFAC-PapersOnLine*, 2017, vol. 50, no. 1, pp. 6918-6927
- [15] B. Kim, Y. Zhang, M. van der Schaar and J. Lee, "Dynamic Pricing and Energy Consumption Scheduling With Reinforcement Learning", *IEEE Transactions on Smart Grid*, 2016, vol. 7, no. 5, pp. 2187-2198
- [16] R. Lincoln, S. Galloway, B. Stephen and G. Burt, "Comparing Policy Gradient and Value Function Based Reinforcement Learning Methods in Simulated Electrical Power Trade", *IEEE Transactions on Power Systems*, 2012, vol. 27, no. 1, pp. 373-380
- [17] Y. Xu, W. Zhang, W. Liu and F. Ferrese, "Multiagent-Based Reinforcement Learning for Optimal Reactive Power Dispatch", *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 2012, vol. 42, no. 6, pp. 1742-1751
- [18] T. Yu, B. Zhou, K. W. Chan, L. Chen and B. Yang, "Stochastic Optimal Relaxed Automatic Generation Control in Non-Markov Environment Based on Multi-Step Q(λ) Learning", *IEEE Transactions on Power Systems*, 2011, vol. 26, no. 3, pp. 1272-1282
- [19] F. Ruelens, B. J. Claessens, S. Vandaal, B. De Schutter, R. Babuka and R. Belmans, "Residential Demand Response of Thermostatically Controlled Loads Using Batch Reinforcement Learning", *IEEE Transactions on Smart Grid*, 2017, vol. 8, no. 5, pp. 2149-2159
- [20] M. Glavic, "Design of a resistive brake controller for power system stability enhancement using reinforcement learning", *IEEE Transactions on Control Systems Technology*, 2005, vol. 13, no. 5, pp. 743-751
- [21] T. Ademoye and A. Feliachi, "Reinforcement learning tuned decentralized synergetic control of power systems", *Electric Power Systems Research*, 2012, vol. 86, pp. 34-40
- [22] H. C. Kilikiran, B. Kekezoglu and G. N. Paterakis, "Reinforcement Learning for Optimal Protection Coordination", *2018 International Conference on Smart Energy Systems and Technologies (SEST)*, Sevilla, 2018, pp. 1-6
- [23] IEEE Distribution System Analysis Subcommittee, "Radial Test Feeders", [Online], 2019, Available: <http://sites.ieee.org/pes-testfeeders/resources/>
- [24] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 2nd Edition, 2018
- [25] V. Minh et al., "Human-level control through deep reinforcement learning", *Nature*, 2015, 518.7540:529
- [26] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang and W. Zaremba, OpenAI Gym, 2016, arXiv:1606.01540
- [27] D. A. S. Jos and S. Elmer, "Typical expected values of the fault resistance in power systems", *2010 IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*, T and D-LA 2010. 602 - 609. 10.1109/TDC-LA.2010.5762944.
- [28] S. Kapoor, "Multi-Agent Reinforcement Learning: A Report on Challenges and Approaches", *Computing Research Repository*, arXiv, 2018, arXiv:1807.09427
- [29] L. Kraemer and B. Banerjee, "Multi-Agent Reinforcement Learning as a Rehearsal for Decentralized Planning", *Neural Computing*, 2016, 190:82-94
- [30] M. Plappert, keras-rl, *GitHub Repository*, [Online], 2016, Available: <https://github.com/keras-rl/keras-rl>
- [31] R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification*, John Wiley & sons, 2nd edition, 2002.
- [32] M. Woodward, "Record U.S. electricity generation in 2018 driven by record residential, commercial sales", *Independent Statistics & Analysis*, [Online], 2019, U.S. Energy Information Administration, Available: <https://www.eia.gov/todayinenergy/detail.php?id=38572>
- [33] IEEE Standards Association, "C37.112-2018 - IEEE Standard for Inverse-Time Characteristics Equations for Overcurrent Relays", *IEEE Standard*, 2019, 10.1109/IEEESTD.2019.8635630