Curriculum Based Reinforcement Learning of Grid Topology Controllers to Prevent Thermal Cascading

Amarsagar Reddy Ramapuram Matavalam, *Member*, *IEEE*, Kishan Prudhvi Guddanti, *Student Member*, *IEEE*, Yang Weng, *Senior Member*, *IEEE*, and Venkataramana Ajjarapu, *Fellow*, *IEEE*

***Abstract*—This paper describes how domain knowledge of power system operators can be integrated into reinforcement learning (RL) frameworks to effectively learn agents that control the grid's topology to prevent thermal cascading. Typical RL-based topology controllers fail to perform well due to the large search/optimization space. Here, we propose an actor-critic-based agent to address the problem's combinatorial nature and train the agent using the RL environment developed by RTE, the French TSO. To address the challenge of the large optimization space, a curriculum-based approach with reward tuning is incorporated into the training procedure by modifying the environment using network physics for enhanced agent learning. Further, a parallel training approach on multiple scenarios is employed to avoid biasing the agent to a few scenarios and make it robust to the natural variability in grid operations. Without these modifications to the training procedure, the RL agent failed for most test scenarios, illustrating the importance of properly integrating domain knowledge of physical systems for real-world RL learning. The agent was tested by RTE for the 2019 learning to run the power network challenge and was awarded the 2^{nd} place in accuracy and 1^{st} place in speed. The developed code is open-sourced for public use. Analysis of a simple system proves the enhancement in training RL-agents using the curriculum.**

***Index Terms*—reinforcement learning, cascading mitigation, actor-critic agents, parallel computing, open-sourced, L2RPN.**

I. Introduction

Grid operators need to ensure that line currents do not exceed physical limits. If left unattended or an appropriate response is delayed, then overloaded lines could lead to cascading due to line thermal limit violations [1]. Transmission system operators (TSOs) prefer an economical and flexible solution like dynamic topology reconfiguration that uses existing infrastructure over the other solutions like load shedding, peak shaving, curtailment, transmission expansion planning [2]–[4]. Even though dynamic topology reconfiguration is preferred by the TSOs [4], it is still beyond the state-of-the-art to optimally control the grid topology "at scale", beyond the level of "transmission line switching" operation [2]. For example, implementation of "bus splitting/merging" operation (node reconfiguration at a substation using the node-breaker model) "at scale" is non-trivial due to the nonlinear combinatorial nature of the graph-like structure of the power grids [5].

[6] proposed an expert system-based approach that incorporates both transmission line switching and bus splitting/merging operations. This expert system-based approach is sufficiently fast but suffers from accuracy issues at times, and also, it cannot account for the impact of an optimal control action over a time horizon [5]. To solve this issue, controllers for dynamic topology reconfiguration are developed to provide optimal control actions over a time horizon. [7] includes the time horizon concept but uses a mixed-integer nonlinear optimization method which takes longer times to solve. [8], [9] proposed a fast method for topology reconfiguration, but due to the problem's large search space, they do not look for the optimal control actions.

Recently, with interest to develop real-time recommendation systems, artificial intelligence (AI) based controllers have been of interest to the industry [4], [5]. AI is used in diverse applications by the industry and few such examples are power grid voltage control [10], stability [11], emergency load shedding [12], energy storage systems [13]. [14] proposes a reinforcement learning-based topology controller but training such a controller to perform well over a wide range of operating scenarios is non-trivial. [5] overcomes such issues by training with more scenarios that resemble real-world behaviors. However, the controllers developed in [5] cannot account for large grids and do not optimize for the line losses.

In this work, a systematic approach to develop topology controllers that plan over a time horizon is proposed. The advantage of the proposed method is that it uses domain knowledge to make the AI-based controller learn well, even in the case of a large solution search space. We propose an actor-critic (A3C) topology controller that can learn by deploying multiple agents in parallel worlds/environments and aggregating the learned policies into a single agent. Furthermore, to simplify the hard-to-solve learning process of developing topology controllers for power grids, we propose power grid domain-specific curriculum learning strategies that can improve any arbitrary AI-based controller's performance and training time.

The contributions of this work are

- A physics-based action-space reduction, state selection and reward design that can enable the learning of topology controllers to prevent thermal cascading.
- A curriculum-learning strategy for accelerating A3C-controller learning with the potential to generalize to other sequential network flow planning problems.
- Testing and validating the proposed curriculum approach on the IEEE 14-bus system and comparing its behavior with a non-machine learning forecast-based agent and an out-of-the-box RL training approach. The proposed method outperforms the other agents because of the domain knowledge embedded in the curriculum strategy.
- Theoretical analysis of curriculum learning on a 3-bus system demonstrating the increased region of convergence for learning the optimal parameter of the RL agent.

A. R. R. Matavalam, K. P. Guddanti and Y. Weng are with the School of Electrical, Computer and Energy Engineering at Arizona State University, emails:{amar.sagar, kguddant,yang.weng}@asu.edu; V. Ajjarapu is with Department of Electrical and Computer Engineering at Iowa State University, email: vajjarap@iastate.edu.

We explain the transfer learning phenomenon from lower levels to higher levels by relating the optimal parameter values and the region of convergence between levels.

We have also open-sourced the code [15] which implements the physics-based curriculum learning along with the physics-based reward function and state selection. The agent learnt by this code placed 2^{nd} in accuracy and 1^{st} in computation speed in the L2RPN-2019 competition [5] conducted by RTE, the French transmission system operator.

The rest of the paper is organized as follows. Section II describes the problem of managing the transmission line congestion and formulates it as a sequential decision-making problem that can be solved by reinforcement learning; Section III describes the general advantage-actor-critic architecture and the training procedure. Section IV describes the challenges in training an A3C grid topology controller and the modified reward to enable agent learning. Section V describes the physics-inspired curriculum-based approach to accelerate the learning of the A3C agent. Section VI presents the simulation results of the trained RL agent using the curriculum approach; Section VII concludes the paper.

II. Problem Description: Managing the Transmission Line Congestion of Power Grids

In this section, first, we introduce the problem of transmission line congestion which causes a cascading event that may result in the blackout of the power grid. Second, to manage such congestion in the power grids, we briefly mention the various preventive techniques and introduce "actions" (real-time topology switching) that are flexible as well as cost-effective from the power grid operator's perspective [2]. Third, we formulate this energy management of power network as a dynamic/sequential planning problem using an objective function and set of constraints. Finally, the complexity of the formulated optimization problem and the size of search space is presented as motivation to "learn" a "policy" (sequence of actions) for the real-time oriented control solution.

A. Black out of power grids due to cascading events

In this subsection, we present the 14 bus system [16] to demonstrate the problem of maximizing the transfer capability of the power grid while avoiding the cascading events over a time horizon. Fig. 1 presents a 14-bus system with 14 substations, 20 transmission lines and 16 injections (both generations and loads combines). In Fig. 1, the substations are indicated by the nodes (blue circles) in the graph; the yellow circles indicate loads, and the green circles indicate generations. Additionally, as shown in the legend of Fig. 1, each substation has two bus bars, namely "bus 1" and "bus 2". An element (either a line or load or generator) can be located at a substation connected to either "bus 1" or "bus 2" (node breaker model).

To represent the realistic power grid operation scenario, realistic generation and load consumption profiles are injected into the power grid for 2000 time-steps of 5 minutes each (equal to 1 week) [17], [18]. Fig. 2 plots the generation and load injection profiles for one scenario. The nomenclature for the generators is gen_{substation_ID}_{gen_ID}, where generators with IDs $0, 1, 2, 3$ & 4 correspond to nuclear, thermal, wind, solar, and hydro-power generation profiles. For

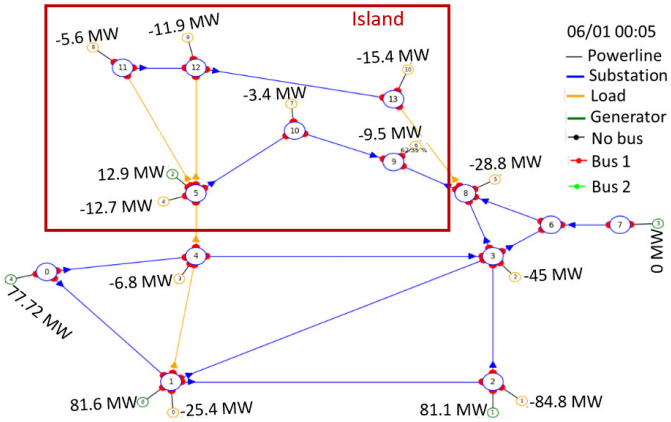


Fig. 1: Power grid cascade event in the 14 bus system due to generation and load injections designed by the L2RPN competition [16], [17].

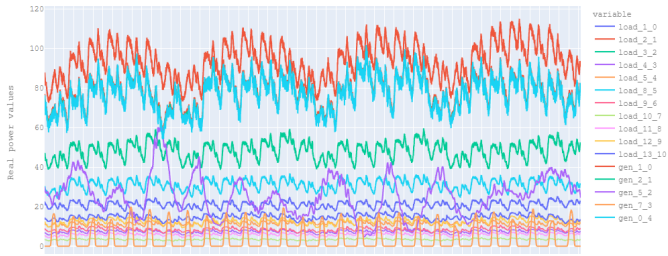


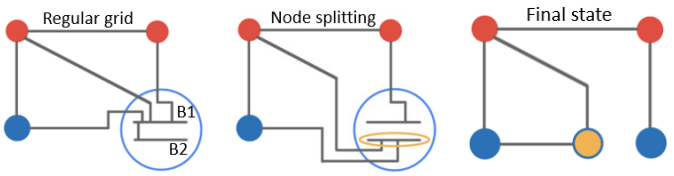
Fig. 2: Generation and load injection profiles in the grid versus time for 2000 time steps of 5 minutes each [17], [18].

example, gen_7_3 has ID = 3 which is the code for a solar plant, and it can be seen that its injection pattern in yellow rises from zero (at dawn) and reaches its peak (at noon) and then drops to zero (at dusk) everyday.

Without performing any modification to the given topology presented in Fig. 1, the injections in Fig. 2 result in a cascading event that leads to power grid blackout. One such cascading event is as follows; first, the transmission line connecting substations 1 and 4 are overloaded and becomes out-of-service. The loss of this line reduces the power grid's overall transfer capability, which in turn overloads the other transmission lines in the power grid. This overloading causes the disconnection of the transmission line $4-5$ two time steps later. Finally, the transmission lines $8-9$ and $8-13$ disconnect simultaneously the next time step due to high line loading of 311.72% and 174.11% respectively, resulting in an island as shown in Fig. 1. However, the formation of islands is not a necessary condition for the blackout of the power grid, and blackout can also occur due to voltage instability condition [16] which is identified by the lack of a solution for a specific set of injections. Hence, it is equally essential to consider cascades that create islands (network flow problem) and voltage stability conditions to ensure power flow solution exists when managing the power flows in the power grid.

B. Topology switching actions

The current focus of the industry is to not only manage power flows in the grid to avoid cascading events as described in Section. II-A, but it is also to maximize the transfer



(a) Power grid with nominal bus bar connectivity.

(b) Bus splitting action using the bus bar connectivity.

(c) Final state of the topology after a bus splitting action.

Fig. 3: Bus splitting using bus bars in a substation [5].

capability of the power grid by minimizing the line losses [5]. Both industry and academia have provided many preventive actions based solutions for congestion management and loss minimization problems. Some of them are new transmission lines (transmission expansion problem), reactive power support, transmission switching, etc. However, installation of new equipment on the power grid is not only expensive, but public acceptance is also a growing concern [5]. Thus, it is preferred to optimize the operation using the flexibility of existing infrastructure. One such method that is both cost-efficient and flexible is the dynamic reconfiguration of grid topology.

The actions required to implement dynamic reconfiguration of grid topology are 1) transmission line switching and 2) bus splitting/merging using the bus bars in a substation. The transmission line switching action involves the decision to make a line in-service or out-of-service. However, bus splitting actions are more complex, and it is explained using a simple 4-bus system from Fig. 3. Fig. 3a presents 4-bus system with five transmission lines and four substations. Each substation in the network has two bus bars to which the power network elements such as loads, generators, transformers, shunt admittances, and transmission lines are connected. Fig. 3a shows a topology with three transmission lines connected to the bus bar 1 (B1) and 2 (B2). For example, as shown in Fig. 3b, a bus splitting action can be triggered to connect two incoming transmission lines to bus bar 2 (B2) and one transmission line to bus bar 1 (B1) separately. This results in a new topology with five nodes, as shown in Fig. 3c, and the new topology can have very different power flow routing properties compared to the original topology.

C. Model formulation: topology controllers for power grids

This subsection discusses the topology controller problem formulation from a traditional optimization approach as a large-scale mixed-integer non-linear programming problem. The difficulty in solving this problem motivates the need for state-of-the-art dynamic optimization techniques. Finally, we provide a concise mathematical representation of the topology controllers for power grids solved in this paper.

The objective of a topology controller for the power grid involves identifying the optimal topology grid configuration (combinatorial) that minimizes the total line loading on the power grid to avoid the formation of islands. This objective must be achieved while ensuring that a power flow solution exists for the optimized grid topology with line currents below thermal limits.

It is shown in [19] that the traditional optimization formulation of identifying optimal topology ***at a given snapshot*** is a large scale non-convex mixed-integer non-linear programming problem (in the interest of space, we did not provide full optimization formulation). This is a computationally intensive optimization problem to solve even for commercial solvers. However, the real-world problem is not a single snapshot problem but rather the optimal topology must be designed considering the variation of load and generator injections **over a time horizon** (several time steps). This significantly increases the computational complexity of the problem. However, there is a need for real-time/fast optimal topology recommendation systems. To address this need, the topology controller problem for power grids is first formulated as a sequential decision-making problem, and then RL agents are trained to solve the problem in real-time using historical data. The sequential planning problem is shown below:

$$\min_{\tau} \sum_{t=1}^{t=n} \sum_{\forall p \in E} \left(\frac{I_{p,t}}{I_{p,max}} \right), \tag{1}$$

$$sub.\ to: f_{\tau}(x_t) = 0;\ \forall t = \{1, 2, \cdots, n\}, \tag{2}$$

$$T(x_t) \in A(T(x_{t-1})); \forall t = \{2, \cdots, n\}, \tag{3}$$

$$I_{p,t} \leq I_{p,max}\ \forall p \in E;\ \forall t = \{1, \cdots, n\}. \tag{4}$$

The aim of the topology controller is to minimize the total line loading on the grid over a time horizon $t = \{1, 2, \ldots, n\}$ (equation (1)) by identifying the optimal topology τ for every time step t with transmission line switching and bus splitting/merging actions. For entire time horizon t, $f_{\tau}(x_t) = 0$ represents the AC power flow constraint of the power grid with different topologies τ and state vectors x_t. The state vector x_t includes the bus voltages, load injections and generator injections and the grid topology representation. The constraint (3) represents the constraint between topologies in consecutive time steps. The grid topology at a time t $(T(x_t))$ should lie in the allowable set of topologies based on the topology at the previous time step $(A(T(x_{t-1})))$. The constraint (4) represents limit on the current magnitude in a transmission line p $(I_{p,t})$. The current must be less than its thermal limits $I_{p,max}$ over the entire time horizon t where $\forall p \in E$ where E is the set of all transmission lines in the power grid.

D. Real-time topology controllers for power grids: size of topology space

The total possible line switching topologies for a grid in Fig. 1 with 20 transmission line is 2^{20}. Similarly, the total bus splitting/merging topologies at a substation with k elements is $\approx 2^{k-1}$ which equal to $1,397,519,564$ unique topologies for the system in Fig. 1. Thus, the total number of possible topology configurations available at a given time step is $1,397,519,564 \times 2^{20}$. Most of these topologies are not viable as they lead to islanding or power flow divergence. *Thus, the complexity of selecting an optimal topology at a given time step is not trivial, let alone computing the strategy over a time horizon*. Hence, there is a need to "learn the strategy" to pick optimal topology (considering the several time steps) rather than exhaustive optimization search methodologies whenever a new grid operating conditions is considered. The field of reinforcement learning deals with learning controllers (also referred to as agents) for sequential decision processes to achieve

a specific objective using techniques from machine learning. The approach to developing such an agent for non-linear sequential planning problems like controlling grid topologies with discrete actions is described in the next section.

III. Deep Reinforcement Learning

This section considers a standard reinforcement learning setup where an agent interacts with a power grid environment ξ over a discrete number of time steps. At each time step t, let the state of the environment be s_t. The agent selects an action a_t from an action set $\mathbf{A}$ which is implemented in the environment ξ. The environment ξ returns the resulting next state s_{t+1} due to action a_t and a reward r_{t+1}. The higher the reward, the better the action a_t corresponds to the state s_t. This procedure is repeated until the environment reaches a terminal state.

The proposed deep reinforcement learning agent uses cooperative actor & critic agents. The actor & critic are represented as deep neural networks with parameters θ, as shown in Fig. 4. The actor-critic architecture is valid for discrete action spaces and is appropriate for node-splitting. The size of the output is equal to the number of discrete actions in the system. Given an action a_{t-1} on an environment ξ, the critic looks at the next state s_t and reward r_t corresponding to action a_{t-1}, it then predicts the value $V(s_t)$ for the state s_t (*policy evaluation*). The actor-network then uses the value $V(s_t)$ and state s_t as inputs into its neural network, and by using the property of the softmax layer, it outputs the probabilities of each action as $\bar{p}(a)$. The action with the largest probability, which is equal to $argmax(\bar{p}(a))$, is selected as the action a_t at time t. This specific action is then implemented in ξ resulting in the next state and reward. During training, the actor uses the feedback of the critic network to update its weights to output higher probabilities for better actions at a given state.

A. Training the actor-critic agent

The objective of an RL agent is to maximize the expected reward overall trajectories τ such that the policy parameter θ optimizes the total reward from the environment. A trajectory is also known as an episode/scenario which constitutes complete gameplay, i.e., a sequence of actions ($policy$) from the initial state to the terminal state. This is given by

$$\begin{aligned}\theta^* &= \underset{\theta}{\operatorname{argmax}} \underset{\tau\sim\pi_\theta(\tau)}{\mathrm{E}} \left[\sum_{t\sim\tau} r(s_t, a_t)\right], \\ \theta^* &= \underset{\theta}{\operatorname{argmax}} \underset{\tau\sim\pi_\theta(\tau)}{\mathrm{E}} \left[r(\tau)\right], \\ \theta^* &= \underset{\theta}{\operatorname{argmax}}\, J(\theta). \end{aligned} \tag{5}$$

The update of policy parameter during the training process at iteration $k+1$ is given by $\theta^{k+1} = \theta^k + \eta \cdot \nabla_\theta J(\theta)$, where η is the learning rate. The efficient learning behavior of the actor-critic network involves the better design of the gradient update of the objective function $\nabla_\theta J(\theta)$.

Policy gradient on objective function: Usually, (5) is solved using gradient descent if the desired objective function $J(\theta)$ is represented as an explicit function. However, in reinforcement learning, the objective function includes the dynamics of the environment, which is a black box. To overcome this drawback, we present the standard REINFORCE update for the policy gradient [20]. [20] shows the derivation of (6) from (5).

$$\nabla_\theta J(\theta) = \underset{\tau\sim\pi_\theta(\tau)}{\mathrm{E}} \left[r(\tau) \cdot \nabla_\theta log\left(\pi_\theta(\tau)\right)\right]. \tag{6}$$

(6) provides the vanilla gradient update equation for a policy gradient neural network-based RL agent. However, this formulation does not provide an efficient learning/optimization algorithm. Specifically, we include a few modifications to (6) in order to make it more efficient by reducing the variance in θ, and discounting future rewards.

Reducing the variance of network weights using the *advantage*: The gradient update using (6) suffers from high variance, which often results in convergence or bad learning of the reinforcement learning agent. It is not reliable or efficient to reduce the high variance by increasing the batch size. To address this, we subtract a constant baseline value $V^\pi(s)$ that is independent of network parameter θ [21] as shown below. This term $(r(\tau) - V^\pi(s))$ is known as $advantage$ value and quantifies the improvement in the total reward for implementing a selected action versus taking no action from the policy network $\pi_\theta(\tau)$ in a given trajectory τ.

$$\nabla_\theta J(\theta) = \underset{\tau\sim\pi_\theta(\tau)}{\mathrm{E}} \left[\nabla_\theta log\left(\pi_\theta(\tau)\right) \cdot \left(r(\tau) - V^\pi(s)\right)\right], \tag{7}$$

where $r(\tau) \neq \sum_{t\sim\tau} r(s_t, a_t)$ but rather the total accumulated reward in a given trajectory discounted such that the actions preformed far away from the current state s_t has minor impact on the reward r_t and similarly the actions implemented in neighborhood of time step t i.e., $\{t-3, t-2, t-1\}$ have a non-zero impact on the reward r_t. This is known as time discounting of the rewards. By simulating N trajectories with fixed neural network weights to approximate the $\mathrm{E}_{\tau\sim\pi_\theta(\tau)}$, the final gradient is given by (8) below where $Q^\pi(s,a) = \sum_{t'=t}^{t_{max}} \gamma^{t'-t} \cdot r(a_{i,t'}, s_{i,t'})$. γ is the discount factor. $t_{max,i}$ is the total time for each trajectory and is the time taken to reach the terminal state.

$$\begin{aligned}\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{t_{max}} \nabla_\theta log\left(\pi_\theta(a_{i,t}|s_{i,t})\right) \cdot \\ \left(Q^\pi(s,a) - V^\pi(s)\right). \end{aligned} \tag{8}$$

The trajectories τ can be simulated in parallel to exploit the multi-core nature of modern high-performance computing hardware. As each of the trajectories is independently simulated, the neural network weights are updated asynchronously. This training approach combined with the actor-critic with regularized advantages leads to the asynchronous-advantage-actor-critic (A3C) [21]. A3C is a conceptually simple RL framework that can be trained in a distributed manner while having the flexibility to address complex problems [21]. We will demonstrate in the results that the curriculum approach shows significant improvement even when using basic RL training methods like A3C. The application of the A3C method to learn grid topology controllers is described in the next section.

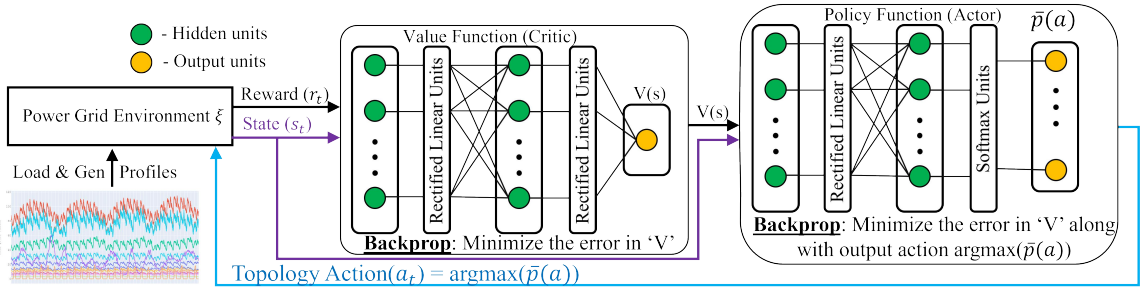


Fig. 4: Neural network architecture for actor critic based RL agent with continuous state and discrete action spaces.

IV. Training A3C Grid Topology Controllers

The python package PyPOWNET [16], developed by the French system operator RTE, is used as the power grid environment to simulate the action of a topology change. The environment uses varying load injection and generation dispatch and the topology resulting from the actions at a time step to estimate the resulting system states such as line flows, number of consecutive time steps in an overload condition, and node voltages. More information on the pypownet package is found in [16]. Each episode consists of load and generation profiles lasting for one unique week from the year 2016.

The default reward from the environment is simple - if the action in the previous time step leads to an unexpected episode termination, then the reward is equal to -1. Otherwise, the return is equal to +1. Unexpected termination occurs when a load/generator is islanded or if the power flow diverges. The islanding can occur due to a bad action in a previous time step or due to line disconnection caused by overloading or by a combination of the two factors. A line disconnection occurs if the actual current exceeds the rating for three consecutive time steps or if the actual current exceeds 1.5 times the rating for a single time step. Thus, the maximum total award occurs when the agent can take actions that make an episode successful for the maximum number of time steps.

We initially trained the A3C agent (Fig.4) on the IEEE 14-bus system using the binary +1/-1 reward using the full state vector as the input to the RL agent. The total number of node splitting actions equals 312, and the state vector size is equal to 438. Any grid topology can be created by a sequence of the node splitting actions, and so the number of actions is significantly less compared to the number of possible topologies mentioned in Section II-D. **We observed that the agent could not continuously operate the grid for more than 50 time-steps even after training for 5,000 episodes. On closer examination of the actions taken by the agent, we realized that the agent was unable to learn effectively** due to the following reasons

- Redundant actions: The equivalence among various node-splitting actions introduces more parameters in the actor neural network slowing its training.
- Correlated and unnormalized states: Correlations are present among the states and their values lie in a wide region as they are not normalized. These attributes lead to ill-conditioned gradients and impede agent learning.
- Unsuitable reward: The binary +1/-1 reward signal is not informative enough for the A3C learning as the agent cannot recognize that the line flows should be maintained below a threshold.

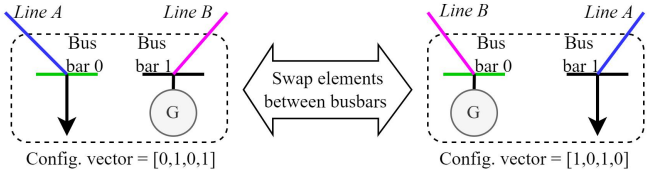


Fig. 5: Two complimentary configurations of a substation containing 4 elements.

Thus, RL agents cannot learn to control the grid topology unless these shortcomings are addressed. In order to overcome the challenge of training the A3C agent, we (i) simplified the action space by logically analyzing the dependence of network topology on the line currents, (ii) reduced the state space by first grouping states based on physics and then analyzing correlations withing the groups, (iii) designed a physically meaningful reward function that is an explicit function of the line current flows. The next subsections discuss the details.

A. Action Space Reduction

The action space for this problem essentially is the connectivity of elements in a substation to the two busbars (bus bar 0 and bus bar 1). Consider a single substation with 4 elements (Line A, Line B, Load & Gen) shown in figure 5. The connectivity of the elements in this substation can be expressed as a 4-d binary configuration vector : [Line A busbar, Line B busbar, Load busbar & Gen busbar]. The figure 5 illustrates two substation configurations [0,1,0,1] and [1,0,1,0].

From the rules of combinatorics, there are a total of 2^4 possible connections for this substation which are the total number of actions available for this substation. However, if we look closely at the two configurations shown in figure 5, it can be seen that the currents in the lines A & B do not change between these two configurations i.e. the current in line A is the same for substation configurations [0,1,0,1] and [1,0,1,0]. The currents are identical because the swapping of elements is same as renumbering the busbars. As the power flow equations are invariant to bus numbering permutations, the resulting line flows under these two actions are the same. Hence, the substation configurations [0,1,0,1] and [1,0,1,0] are complementary.

As thermal cascading is triggered by line currents, we only need to consider one of the complementary actions as the impact of the complementary action is identical from a cascading perspective. For every substation configuration, we can find a

This article has been accepted for publication in IEEE Transactions on Power Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPWRS.2022.3213487

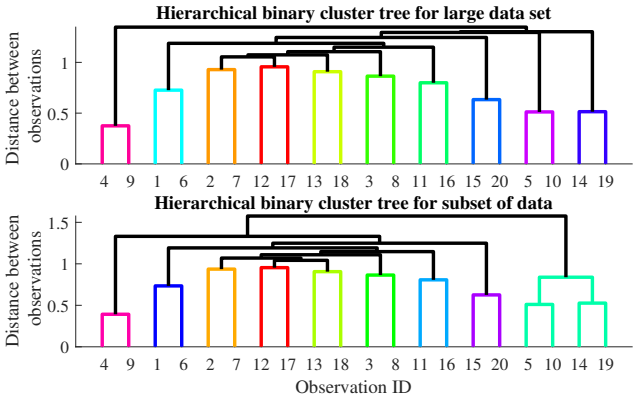


Fig. 6: Dendrogram with clusters of the observations of the current & forecast, active & reactive powers for 5 loads. Obs 1-Obs 10: Active power. Obs 11-Obs 20: Reactive power.

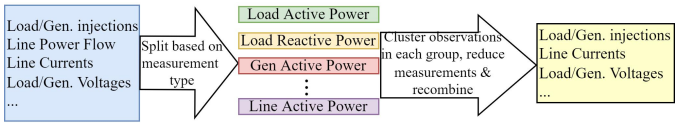


Fig. 7: Flowchart of the state space reduction approach

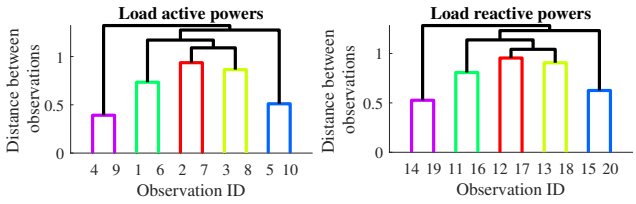


Fig. 8: Dendrogram with clusters of the observations of the current & forecast active (left) & reactive (right) powers for 5 loads using the reduced subset of data. Obs 1-Obs 10: Active power. Obs 11-Obs 20: Reactive power.

complimentary configuration by swapping '0's and '1's and this complementary configuration is a redundant action due to the same reasoning. Thus, the pairing of complementary actions is true for any substation with any number of elements. We select only one of the two complementary configurations as valid actions that are output by the RL agent and thus the total number of actions possible can be halved, **reducing the total number of node splitting actions from 312 to 156**.

B. State Space Analysis and Reduction

The observations from the environment are used as inputs by the RL agent to determine the action in the next time step. The full observation vector of the environment has detailed information about the status of the grid such as - (i) Topology information such as the busbar that each load/gen./line connects to in a substation along with the line status (ii) Power flow information such as load/gen. injections, line flows, and load/gen. voltages (iii) Forecasted load powers, generator powers & voltages. The total number of observations at each time step is equal to 438. The large number of the observations increases the number of parameters in the actor and critic neural networks (NNs). Furthermore, there is significant correlation among various observations. The large number of NN parameters and correlations impede the learning of the RL agent. Thus, to improve RL-agent learning, a key step is to choose a subset of the observations that retain the key information about the system with minimal correlations. As the observations at a time step are a function of the actions at the previous time step, the observation subset chosen should be able to uniquely identify the grid topology and line flows under the many actions possible by the agent. This will be the state vector used as input for the training and evaluating the RL agent.

Traditionally, data-analytics techniques such as clustering [22] are used to reduce the number of observations from correlations. However, applying these approaches to the full 438-dimensional observations for a few random actions will lead to spurious correlations and inappropriate clustering. A large dataset resulting from many actions is necessary to fully characterize the correlations due to the large size of the observation vector [22] and the action space, making the state space reduction a difficult task. To illustrate this, consider the current and forecasted active and reactive injections for 5 loads - a total of 20 observations. The hierarchical clustering results using the Ward method [22] for these observations with a large and small dataset is presented in Figure 6. A normalized cutoff distance of 1 (along y-axis) is used to identify the clusters. Clustering using the large dataset splits the observations into 10 clusters, with each cluster consisting of the current and forecasted active/reactive powers. However, for the clustering using a smaller dataset, a spurious correlation between active power at load-5 and reactive power at load-4 causes them to be clustered together along with their forecasted observations. These spurious correlations are likely to be present when clustering high dimension data [22] and so a more physically meaningful state-space reduction approach is necessary.

This challenge is addressed by leveraging the physics of the grid. We first split the observations into groups corresponding to the physical quantity measured. Then the observations within each group are separately clustered and only one measurement from each cluster is used. This approach is illustrated in Figure 7. The clustering result on applying this approach on the reduced observation data for the current and forecast active and reactive injections for 5 loads is shown in Figure 8. Even though this clustering is derived from a smaller data-set, it is same as the clustering derived from large data in Figure 6, demonstrating the advantage of physics-informed clustering approach to perform state space reduction.

We further reduce the states by leveraging the power flow equations that relate the active and reactive power flows in a line to the grid topology, bus power injections and bus voltages. Thus, the line active and reactive power flows do not need to be monitored explicitly if the element connectivity information, bus injections & voltages are used. The final states used as input to the RL agent are:

- Line status & busbar to which each element is connected.
- Normalized current magnitude and current direction.
- Generator active power and voltage dispatch.
- Load active power and reactive power demand.
- Time-steps before a substation can be controlled.

These final chosen states are then individually scaled so

that the maximum values of all the states are in the same order of magnitude which ensures numerical instabilities are not present during the training of the RL agents. **The final size of the state space is reduced from 438 to 162 - a reduction of more than 2.5x. Reducing the number of actions and states significantly cuts the number of NN parameters θ from $\sim 110k$ to $\sim 50k$.**

C. Designing a Suitable Reward

As the RL agents are trained to maximize the total reward over a time horizon, the reward needs to be appropriately defined to reflect the user's intention for the agent. As the grid controllers should maximize the duration of grid operation, the binary +1/-1 reward seems to be sufficient as the maximum cumulative reward ($J(\theta)$) occurs when the agent avoids termination and increases the number of successful steps. However, the discrete reward leads to a difficult optimization problem as the gradient of $J(\theta)$ can be zero over large regions. Furthermore, the reward at a particular step is independent of the state and does not provide any information about the risk of cascading.

As the underlying mechanism for cascading and thermal disconnections is the line current exceeding its limit, a reward function that explicitly uses the line currents and limits is preferred. Further, this reward function should be designed to discourage line overloading and scenario termination. Thus, we define a new reward function, $r(s_t)$, shown in (9), with these properties. The reward is essentially the sum of line margins for all the lines if no unexpected termination occurs and is a large negative value (-100) if the termination occurs due to islanding or divergence.

The function $R(x)$ (shown in Fig.9) is a proxy for the line margin and its value reduces as the line loading increases. It is negative if the line current is greater than 0.95 times its maximum current limit. A threshold of 0.95 is used instead of 1.0 during training to 'robustify' the A3C agent and aids in generalizing the A3C agent to similar but unseen states. The value of α in (9) determines the penalty for a line overload. Fig.9 plots the function $R(x)$ for varying values of α. The explicit dependence of the reward on the state facilitates the value function ($V^{\pi}(s)$) learning which in turn makes the learning of the policy π easier. This reward is maximum when the A3C agent has the maximum number of continuous successful time steps for all training scenarios with the least line usage, leading to the same outcome of the binary reward. Thus, the solution of (5) using the modified reward is also a solution to the (5) using the binary reward.

$$r(s_t) = \begin{cases} \sum_{\forall p \in E} R\left(\frac{I_{p,t}}{I_{p,max}}\right); & \text{if not terminal time step} \\ -100; & \text{if terminal time step} \end{cases} \tag{9}$$

$$R(x) = \begin{cases} (0.95 - x); & x \leq 0.95 \\ \alpha \cdot (0.95 - x), \alpha \geq 1; & x > 0.95 \end{cases} \tag{10}$$

The identification of bottlenecks impacting the A3C agent learning and developing mitigation strategies resolving them by exploiting the physical understanding of the grid is the first main contribution of the paper. *This behavior of the reward function was conveyed to RTE and disseminated to a wide audience. As a result, multiple participants of the L2RPN-2020 challenge used this reward function, including the winners.* Implementing the above strategies enables the A3C agent to operate the 14-bus system for more than 200-time steps for a few scenarios after training for 5,000 episodes. However, the learning was very slow, and most of the scenarios failed after 100-time steps. Based on the states and actions analysis, we observed that the agent needs many episodes to learn to avoid risky actions. This is because the number of time steps the environment can operate in a risky topology is typically very low (< 5 time steps) before causing line overloads leading to episode termination. Thus, standard RL training approaches are not designed for learning topology controllers effectively. To address this drawback and accelerate the learning, we developed an efficient training method using physics-inspired curriculum learning to obtain high accuracy for the RL-based topology controllers. This is explained in the next section.

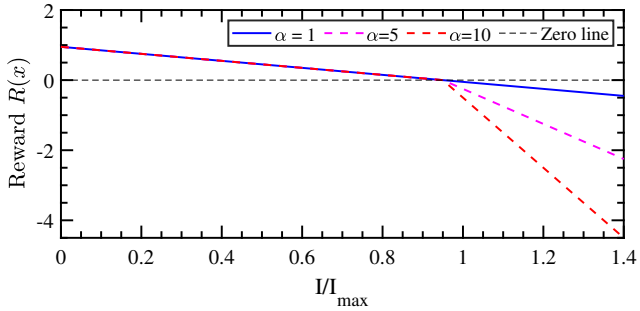


Fig. 9: Plot of the function $R(x)$ versus the line current for various parameters of α.

V. Physics Inspired Curriculum Strategy for Accelerated Learning

Curriculum learning is the idea that neural networks learn a difficult task most effectively when first trained on a simpler task. Curriculum learning is inspired by how humans learn - initially learning simple concepts before attempting complex tasks. It is a form of transfer learning as solving simple tasks is leveraged to solve the more complicated task. A proper curriculum (sequence of tasks with increasing hardness) should be designed to apply this approach for effectively learning grid controllers. Designing an effective curriculum is not easy, and a bad curriculum can impede agent learning. Recent approaches [23] have proposed to learn curriculum strategies as a part of the overall ML-based approach for classification or regression tasks. In recent years, curriculum learning to accelerate training of RL agents has been explored in various settings [24]. However, it has not been explored for controlling network flows. **In this section, we present the physics-inspired curriculum using the behavior of network flows and cascading. The designed curriculum accelerates the A3C agent learning and is the second contribution of the paper. As far as the authors are aware, this is the first time a curriculum has been designed to train RL agents to control network flows.**

There are a few settings in PyPOWNET that indirectly can increase or reduce the hardness of the environment, as seen by the agent. These configuration parameters are:

- The soft overload threshold (SOT), which is the fraction $\frac{I_{p,t}}{I_{p,max}}$ beyond which an overload alarm is triggered.

This article has been accepted for publication in IEEE Transactions on Power Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPWRS.2022.3213487

TABLE I: Environment parameters for the curriculum levels.

Level	Reward α	SOT	COL	HOT
1	1	10^9	10^9	10^9
2	5	2	15	10^9
3	10	1	3	1.5

- The consecutive overload limit (COL) which determines how long a line can be continuously in soft overload before the line is disconnected.
- The hard overload threshold (HOT) which is the fraction $\frac{I_{p,t}}{I_{p,max}}$ beyond which line p immediately disconnects.

The default parameters of the environment are SOT = 1.0, COL = 3 time steps and HOT = 1.5. These parameters imply that the overload counter is triggered when the line current exceeds its rating, and the line will be disconnected if the current remains continuously above the SOT limit for 3 steps (COL). If the line current exceeds 1.5 times the rating (HOT), then it is immediately disconnected. Cascading line outages are the main reason for unexpected termination due to the RL agents, and so initially, we need to prevent cascading in the environment. **As the problem of cascading occurs due to a sequence of lines disconnecting due to overloads, relaxing the line limit enforcement will directly prevent cascades. It is important to emphasize that the line limits ($I_{p,max}$) are not modified in any of the levels, only the enforcement of the limits is relaxed. Thus, the reward will be negative if the line limits are exceeded. This negative reward will discourage the A3C agents from taking actions that cause line overloads even if the line limit is not enforced.**

The designed curriculum consists of three levels with increasing difficulty. The environment parameters for the three levels are shown in Table I. The α parameter used in the reward function is also increased to ensure that the penalty for overload increases at higher curriculum levels. In Level-1, the SOT is very large (10^9), which implies no line limit enforcement. In level-2, the line disconnections are enforced with a large COL of 15. The HOT is very large for this level which prevents immediate line disconnection. Level-3 corresponds to the default environment behavior described above. The levels are designed in a sequential manner that gradually increases the 'strictness' of the enforcement. Next, three propositions are discussed that provide the rationale for improved agent learning with the designed curriculum.

A. *Proposition 1: More training samples are seen by the agent for lower levels than higher levels*

The relaxed line limit enforcement allows the operation of the grid for more time steps in an episode and generates more samples for training the agent at lower levels. Consider the situation shown in Fig 10 displaying the normalized line current in a line for three-parameter values. An agent with any of the three parameters will see the entire scenario in level-1. The training samples for level-1 include samples where the agent's actions led to unfavorable/risky states. However, in level-2 an agent with the parameter θ_1 will cause the environment to terminate 15 time-steps after t_1. A similar case occurs for θ_2 at t_3. Hence, there are lesser samples from unfavorable/risky states with level-2 and level-3 enforcement.

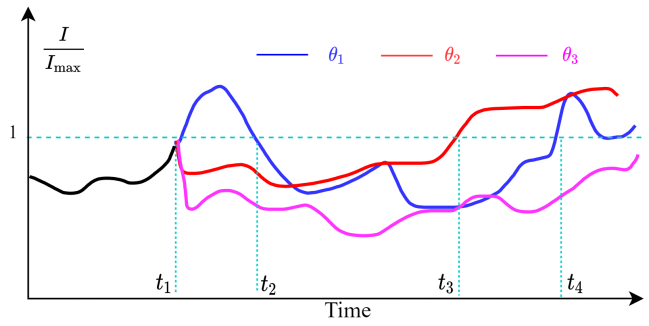


Fig. 10: Conceptual plot of the normalized line current versus time for three parameter values in a single scenario.

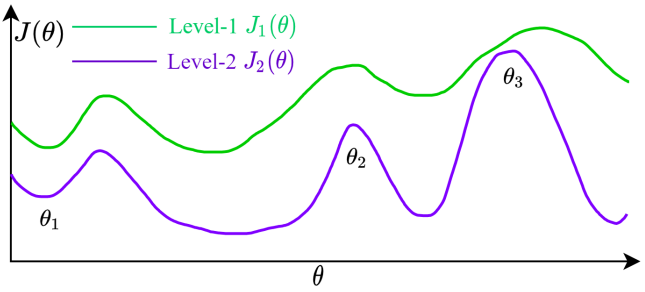


Fig. 11: Conceptual behavior of the objective function $J(\theta)$ versus varying parameters θ for different curriculum levels.

B. *Proposition 2: Agent learning for lower levels is easier than higher levels as the function $J(\theta)$ is smoother*

Local maxima in the objective function impede the agent learning as they prevent gradient-based methods from escaping them. Curriculum learning smooths out the objective function and makes it easier for the optimization approaches to escape the local maxima. Consider Fig. 11 which plots the conceptual objective function for two levels assuming one dimensional θ. $J_2(\theta)$ has sharper peaks and troughs due to the fact that the enforcement of line currents is strict. Based on proposition 1, the relaxation of constraint enforcement in level-1 leads to a larger region of the parameter space where the agent performs well. The performance enhancement is largest for parameters that performed poorly in level-2 as they have the most room to improve (θ_1 & θ_2 in this case). The performance of θ_3 does not improve musch as it is very successful in level-2. Hence, the overall effect on $J(\theta)$ is to reduce the variation between the peaks and troughs. Thus, $J(\theta)$ is smoother for level-1(level-2) than level-2(level-3) over a larger parameter set, easing the agent learning in lower levels compared to higher levels.

C. *Proposition 3: Agents trained on lower levels perform well on higher levels as well*

The region in the parameter space which maximizes the objective $J_1(\theta)$ is close to the optimal value for the objective for $J_2(\theta)$. This is due to the design of the reward function, which penalizes overloads. An agent trained for sufficient episodes in level-1 will prevent large line overloads and avoid actions that island a part of the grid. Thus, an agent trained on level-1 for a level of success is likely to perform well on level-2 as well. Using the agent trained on level-1 as an initial agent for level-2 has the advantage of transferring learning

TABLE II: Mapping of physical components of networked infrastructure to a graph along with the overload phenomenon

Network	Graph Node	Graph Edge	Overload Phenomenon
Electric	Substation	Lines	Current flow in lines
Gas	Compressors	Pipelines	Gas flow in pipelines
Internet	Routers	Fiber Optics	Data flow thru routers
Interstate	Junctions	Roads	Traffic flow on roads

from a simpler level as it does not need to relearn some action sequences. This property leads to efficient learning. The same logic holds while using an agent from level-2 to level-3. Hence, learning the agent sequentially from level-1 to level-3 ensures that the knowledge learned by the agent for 'easier' levels is retained, and the agents need fewer scenarios to satisfy the 'stricter' constraints of a 'harder' level.

Hence, gradually increasing the level during the learning will lead to accelerated learning of the A3C agent because of the above three prepositions as follows: (1) it can observe and learn from more samples in lower levels and (2) gradient steps are more likely to skip the local maxima in the objective function in the relaxed levels (3) effective transfer learning occurs due to the design of the reward function. **Theoretical analysis and proof of the three propositions for a simple system is described in the appendix.** The curriculum approach is very much related to continuation and homotopy methods [Section 3 in 23], [25]–[27] and can be understood by an analogy to the power flow problem. Solving power flow for stressed systems is sensitive to initial conditions. Continuation/homotopy methods [25], [27] overcome the challenge by solving a low load scenario and gradually increase the loading just like increasing the levels from 'easy' to 'hard' for cascading. Next, we discuss how the curriculum can be adapted to prevent overload cascading in other networks.

D. Justification for extending curriculum to prevent cascading in other networks

The curriculum design and the three propositions discussed in the previous subsections only used the properties of overloading in power networks (flow in the lines) after an outage. The curriculum design does not explicitly use any other feature of power grids such as the power flow equations. There is significant literature in network sciences [28], [29] that demonstrates that the overload cascading model shares very similar features in many physical infrastructures that can be represented as graphs such as electric networks [30], computer networks [31], natural-gas networks [32] and transportation networks [33]. Further proof of the close analogy between the cascading phenomenon in these disparate networks is the scale-free power law that governs the distribution of the cascading sizes in power grids [34] and transportation networks [35]. The mapping between the physical components for a few infrastructure networks to the graph nodes and edges along with the overload constraint is presented in Table II. Hence, the curriculum design can be translated to learn agents that prevent cascading overload failures in other infrastructure.

A node/edge overloading in these networks causes the failure of the component and the redistribution of the network flow that can lead to overloading and failure of the remaining nodes/edges. This is a generalization of the cascading failure propagation which we described in section II for power grids with line-limits. For example, computer networks, which are node-limited, can exhibit cascading when routers stop responding due to an excessive number of data packets. The data packets are then redirected to other routers that can also overload and fail, leading to cascading [28], [31].

The curriculum design relaxes the overload constraints on the limiting components of the network and penalizes overloads in the reward function design. **Thus, the curriculum described in the section can be applied to prevent cascading in various networks due to component overloads arising from flow reconfiguration.** For example, in the case of computer networks, we can allow an overloaded router to continue operating with a penalty dependent on the amount of data overload it observes. The penalty (α) will be initially low and will be increased as training progresses. Thus, training RL agents in these domains to prevent cascading by optimizing network flow can benefit from the proposed network physics-based curriculum based on the overload phenomenon.

VI. Simulation Results

In this section, results on the IEEE 14 bus system are presented. The load & generation scenarios are taken the pypownet package [16] as a part of the Learn to Run the Power Network (L2RPN) 2019 challenge [5].

A. Agent Training and Evaluation Setup

Deep neural networks represent both the actor and critic with two hidden layers of sizes 200 and 50. The first layer of the neural network is shared between the actor and critic leading to joint training of the A3C agent. The learning rate for the actor is 0.0005, and the learning rate for the critic is 0.001. A discount factor (γ) equal to 0.95 is used to calculate the time discounted rewards for the training. A total of 50 unique training scenarios are selected from the dataset, and 50 threads are used in parallel during the A3C training procedure. Each unique scenario is made up of 2000 time steps of 5 minutes each that corresponds to 1 week of operation. An agent that continuously operates the grid for all time steps in a scenario is categorized as a successful agent for that scenario.

The following agents are used to verify the utility of reinforcement learning and curriculum learning to address the topology problem. There is no training in the forecasted power flow-based agent, as it is a brute-force approach, while the A3C agents are trained for 30,000 episodes on the 50 unique scenarios.

- Forecasted power flow (FPF) based agent: This is the non-machine learning approach in which the forecasted injections at the next time step are used to identify the best action at a given time step. This approach is a 'greedy' approach as it is based only on a single-step forecast. It cannot account for how an action would change the line currents further into the future.
- Baseline A3C (A3C) Agent: This agent is trained on level-3 enforcement, the hardest level, using the modified reward with action/state-space reduction and state normalization throughout the training process.
- Curriculum A3C (CA3C) Agent: This agent is trained using the curriculum strategy presented in Section V along with the modified reward with action/state-space

This article has been accepted for publication in IEEE Transactions on Power Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPWRS.2022.3213487

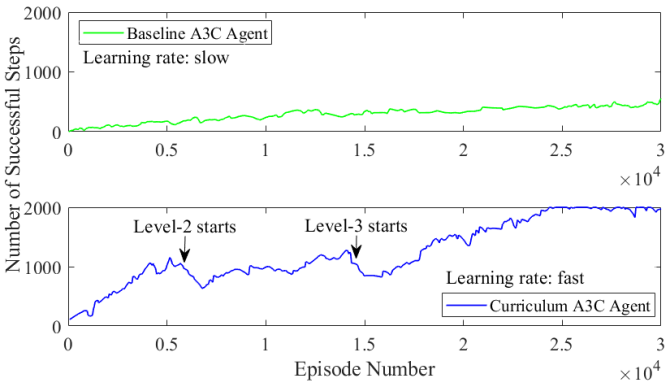


Fig. 12: Plot of the number of successful steps versus the episodes during training of A3C and CA3C agents.

reduction and state normalization. The transition between the levels occurs when the agent can continuously operate the grid for >1000 time-steps on at least 25 scenarios.

B. Training of the A3C and CA3C Agents

The agents are implemented in Keras and are trained using TensorFlow for 30,000 episodes and the code is available on GitHub [15]. The number of successful time steps at each training episode for the two agents is shown in Fig. 12. The median of successful time steps for each of the 30,000 episodes over a window of 15 different scenarios/weeks is plotted in Fig. 12 to smooth out the large variation among the episodes. The enforcement level of CA3C is initially level-1. Based on the agent's performance, the enforcement level is increased to level-2 at episode 6000 and increased to level-3 at episode 14000. The agent at these episodes is saved for the further analysis presented in subsection-D.

It can be seen from the plot in Fig. 12 that the learning is comparatively slow for the A3C agent. The number of successful steps of the A3C agent in the training phase at the end of 30,000 episodes is around 500 steps. For the CA3C agent, there is a much faster training rate as the number of successful steps increases quickly. This is because the enforcement level is low (level-1). As soon as the level is increased after 6k episodes, there is a drop in the number of successful steps. After a few more episodes, the learning algorithm will update the network parameters appropriately and improve them till the next level is enforced at 14k episodes. The same temporary drop in performance can be seen after 14k episodes. It was observed that the variance of the rewards observed during A3C training is higher compared to CA3C training. This is due to the 'rough landscape' of the objective function $J(\theta)$ for level-3 enforcement.

C. Evaluation of Various Agents on Test Scenarios

In this section, the performance results of the various agents are presented and analyzed. The three (FPF, A3C, and CA3C) agents are evaluated on 150 test scenarios enforced at the hardest level, with each scenario lasting 2000 time steps. The agents are used to identify topology actions only during the time-steps when the current flow in at least one line exceeds 80% of its limit. The agents are scored on each scenario based on the number of continuous successful time steps before the scenario terminates due to islanding or system divergence. This information is plotted in Fig. 13.

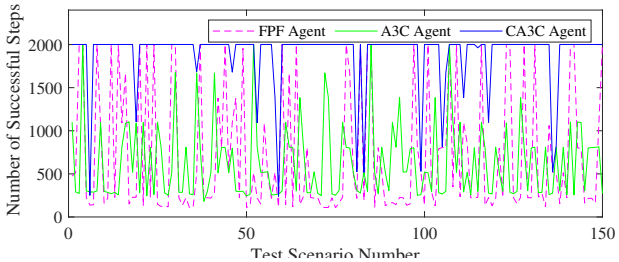


Fig. 13: The number of successful steps for the various agents in each test scenario with level-3 enforcement. Each scenario is a unique one week worth of operating conditions.

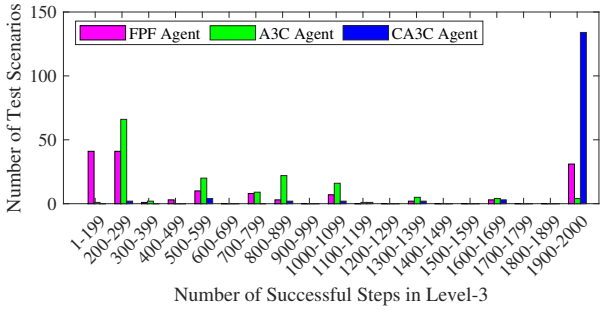


Fig. 14: Histograms of the successful steps for the various agents for 150 test scenarios with level-3 enforcement.

The performance of the A3C agent is poor as only a few of the scenarios successfully reached the end. Most of the scenarios with the A3C agent terminated within 500 time-steps. In contrast, the performance of the CA3C agent is much better as most (120 out of 150) of the scenarios successfully reached the end. The behavior of the three agents is summarised in Fig. 14 which plots the histogram of the number of successful time steps for each test scenario partitioned into bins of 200-time steps. The superior performance of the CA3C agent can be clearly seen from this histogram. These results demonstrate that (i) reinforcement learning agents can perform better than a single look-ahead non-ML-based approach on systems with complex constraints on the actions (ii) The CA3C agent performs significantly better than the A3C agent trained without a curriculum.

D. Illustrating Accelerated Learning Due to Curriculum

To demonstrate why the proposed approach-based curriculum training leads to a better agent, we analyze the behavior of the A3C agents on the test scenarios with level-1 enforcement. In addition, we also evaluated the behavior of the CA3C agents stored when the enforcement levels were raised. CA3C-6k is the agent when the curriculum transitions from level-1 to level-2 at episode 6000, and CA3C-14k is the agent when the curriculum transitions from level-2 to level-3 at episode 14000. We recorded the normalized line currents for all the lines for each agent and scenario, leading to a large database. A box-whisker plot is used to present the statistics of the dataset visually and is shown in Fig. 15. The red line within each box is the median current flow for all test scenarios for a particular A3C agent. The top and bottom boundaries of the box correspond to the inter-quartile range of the current for each line. The top and bottom whiskers correspond to the

This article has been accepted for publication in IEEE Transactions on Power Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPWRS.2022.3213487

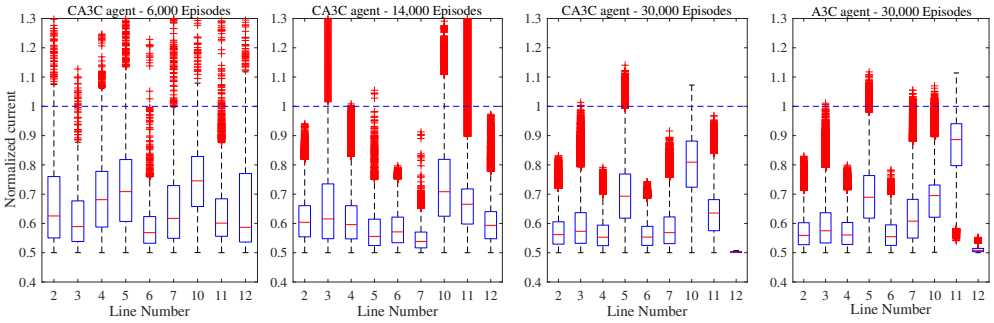


Fig. 15: Comparison between the boxplots of the normalized line currents in each line due to the actions of the various A3C agents for all the time steps in the 150 test scenarios with over-current cascading disabled. Normalized current = I/I_{max}.

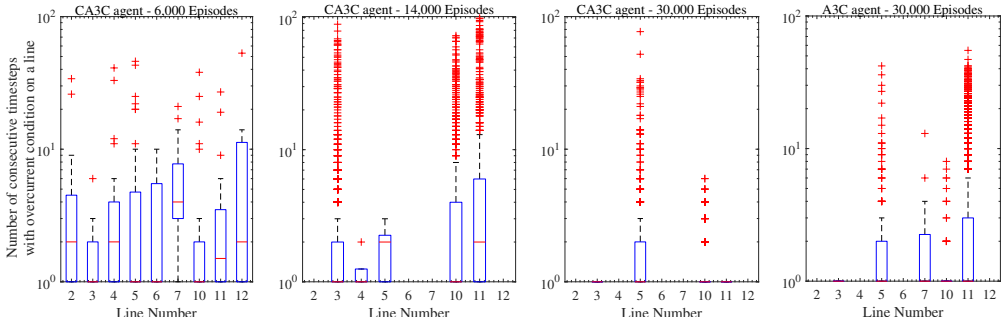


Fig. 16: Comparison between the boxplots of the number of consecutive time-steps for over-currents in each line due to the actions of the various A3C agents for all the time steps in the 150 test scenarios with over-current cascading disabled.

estimated maximum and minimum values of the current for each line without the outliers. The outliers of the current for each line are represented as red crosses.

By observing the box plots in Fig. 15, it can be seen that the CA3C-6k agent has many samples with overload. This is expected as the agent has not yet learned fully to reduce the line currents. Instead, the agent has prioritized the identification of action sequences that would lead to termination due to islanding without any cascading. In the box plot of the CA3C-14k agent, we can observe that the agent has learned to reduce the line current below the maximum value for many of the lines. This is exactly the intention behind increasing the value of α and adding a time delay to the overload disconnection. Finally, after 30,000 episodes, the CA3C agent reduces the overload to just 2 lines. In contrast, the A3C agent after 30,000 episodes has overloads in 4 lines, making it more susceptible for cascading.

However, this analysis does not fully explain the poor performance of the A3C agent, as most of the lines have avoided overloading. This is because the number of time steps that a line is continuously overloaded ($t_{p,over}$) is the actual reason for line disconnection, and this is not the same as the total number of time steps that a line is overloaded. For example, an agent that can immediately rectify a line overload in one step will have a $t_{p,over}$ equal to 1. Thus, a successful agent has smaller values ($t_{p,over}$) for all lines. The data set is analyzed, and $t_{p,over}$ is calculated. The resulting $t_{p,over}$ are plotted in Fig. 16.

The box plots in Fig. 16 demonstrate that the time spent by each line continuously in overload reduces as the CA3C agent learns. Initially, many lines have high $t_{p,over}$. As the learning progresses, the value of $t_{p,over}$ reduces. At the end of 30,000 episodes, the CA3C agent can limit $t_{p,over}$ to 1 for all lines except line-5. These values of $t_{p,over}$ are low enough that most overloads do not cause line disconnections, limiting the impact of most of the overloads. In contrast, the maximum inter-quartile value of $t_{p,over}$ for the final A3C agent is equal to 3 for line 5, 4 for line 7, 1 for line 10, and 8 for line 11. These values are much larger than the HOT, and thus they will lead to cascades in most test cases. This is exactly what we observe in Fig. 13 for the A3C agent.

Hence, the CA3C agent can minimize the overloading occurrences and also reduce the continuous-time in the over-loaded state, thus leading to improved performance compared to the A3C agent. **The efficient learning of the CA3C agent is verified on the IEEE 14 bus system by systematically analyzing the cause of the failure of the partially trained CA3C agents and the A3C agent. The statistical analysis of the line currents and consecutive duration of the line overloads is used to justify the gradual improvement in the performance of the CA3C agent as training proceeds. This is the third contribution of the paper.**

E. Agent Performance in L2RPN-2019 Competition

The A3C agent trained using the curriculum presented in this paper placed 2nd in the L2RPN-2019 challenge by RTE. RTE tested the trained CA3C agent on hidden scenarios of varying length, and the agent was successful in all the cases. RTE's analysis from [5] for the trained CA3C agent mentions that the agent is quite stable due to its small action space but has the ability to go back and forth, illustrating the impact of using the physics of the system in the designing of the action space. **The authors have open-sourced the code to train the**

A3C agent with the physics-based curriculum approach for controlling grid topology on GitHub from [15]. The novel approach (state-space reduction, action space reduction, modified reward, & curriculum training methodology) and the corresponding code has been already used by another team to win the L2RPN-2020 challenge on a larger system [36].

VII. Conclusion and Future Work

This paper describes how domain knowledge of power system operators can be integrated into reinforcement learning frameworks to effectively learn agents that control the grid to prevent cascading through grid reconfiguration. The non-linear and combinatorial nature of the grid reconfiguration problem means that no existing optimal power flow based approach can tackle this problem. We have developed a training approach for RL agents that has successfully operated the grid under various test scenarios. The key to training is to incorporate the knowledge of power system operation into various aspects of the reinforcement learning framework.

First, we reduce the action space and the state space dimensions reduce by analyzing the grid topology and operating conditions. Next, a reward function is designed to provide gradients to improve the RL agent even when the lines in the grid have overloaded. Finally, an effective physics-based curriculum approach is incorporated into the training procedure through environment modifications that enables the agent's accelerated learning. The learning procedure is stabilized and made robust to the natural variability in grid operations by employing a parallel training procedure that trains on multiple scenarios of the power grid at the same time. This training procedure reduces the sampling bias that is likely to deteriorate performance when training in a sequential. Without these enhancements to the training procedure, the RL agent failed for most test scenarios, illustrating the importance of properly integrating domain knowledge of the physical system for RL learning for a real-world system. The developed code is open-sourced [15] for use by researchers and utilities. The agent was tested by the French transmission system operator, RTE, in the 2019 learning to run the power network challenge and was awarded the 2^{nd} place in accuracy and 1^{st} place in computation speed.

Our next steps are to increase the scalability of the approach and to apply the method to (a) larger power grids with additional actions such as generator and load controls and (b) other networked infrastructure demonstrating cascading. We plan to utilize graph based neural networks to increase the generalization capability of the RL agents. Further, we will leverage recently developed RL frameworks such as RLzoo [37] that include advanced RL algorithms such as soft-actor-critic (SAC) and distributed proximal policy optimization (DPPO) which efficiently handle continuous and discrete action spaces for large problems. Finally, we will explore merging model predictive control based approaches with this framework and increase the confidence of power system planners and operators in these agents. More theory will be developed for the curriculum design and deeper analysis will be performed to improve the explainability of the approach.

References

[1] P. Hines, K. Balasubramaniam, and E. C. Sanchez, "Cascading failures in power grids," *IEEE Potentials*, 2009.

[2] E. B. Fisher, R. P. O'Neill, and M. C. Ferris, "Optimal transmission switching," *IEEE Trans. on Power Systems*, 2008.

[3] M. Soroush and J. D. Fuller, "Accuracies of optimal transmission switching heuristics based on DCOPF and ACOPF," *IEEE Trans. on Power Systems*, 2013.

[4] E. Karangelos and P. Panciatici, "'cooperative game'inspired approach for multi-area power system security management taking advantage of grid flexibilities," *Phil. Trans. of the Royal Society A*, 2021.

[5] A. Marot, B. Donnot, C. Romero, B. Donon, M. Lerousseau, L. Veyrin-Forrer, and I. Guyon, "Learning to run a power network challenge for training topology controllers," *Electric Power Systems Research*, 2020.

[6] A. Marot, B. Donnot *et al.*, "Expert system for topological remedial action discovery in smart grids," *IET Digital Library*, 2018.

[7] G. Granelli, M. Montagna, F. Zanellini *et al.*, "Optimal network reconfiguration for congestion management by deterministic and genetic algorithms," *Electric Power Systems Research*, 2006.

[8] G. Schnyder and H. Glavitsch, "Integrated security control using an optimal power flow and switching concepts," *IEEE Trans. on Power Systems*, 1988.

[9] G. Schnyder and H. Glavitsch, "Security enhancement using an optimal switching power flow," *IEEE Trans. on Power Systems*, 1990.

[10] J. Duan, D. Shi, R. Diao, H. Li, Z. Wang, B. Zhang, D. Bian, and Z. Yi, "Deep-reinforcement-learning-based autonomous voltage control for power grid operations," *IEEE Trans. on Power Systems*, 2019.

[11] S. You, Y. Zhao, M. Mandich, Y. Cui, H. Li *et al.*, "A review on artificial intelligence for grid stability assessment," in *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, 2020.

[12] J. Li, S. Chen, X. Wang, and T. Pu, "Research on load shedding control strategy in power grid emergency state based on deep reinforcement learning," *CSEE Journal of Power and Energy Systems*, 2021.

[13] S. Wang and et. al., "Deep reinforcement scheduling of energy storage systems for real-time voltage regulation in unbalanced LV networks with high PV penetration," *IEEE Trans. on Sustainable Energy*, 2021.

[14] M. Subramanian, J. Viebahn, S. H. Tindemans, B. Donnot, and A. Marot, "Exploring grid topology reconfiguration using a simple deep reinforcement learning approach," in *IEEE Madrid PowerTech*, 2021.

[15] A. R. R. Matavalam, K. P. Guddanti, Y. Weng, and S. Indela, "L2RPN IJCNN 2019 Competition - Second Place Solution." https://github.com/amar-iastate/L2RPN-using-A3C, 2019.

[16] M. Lerousseau, "Design and implementation of an environment for learning to run a power network (L2RPN)," *arXiv preprint arXiv:2104.04080*, 2021.

[17] B. Donnot, "Grid2op- A testbed platform to model sequential decision making in power systems." https://GitHub.com/rte-france/grid2op, 2020.

[18] B. Donnot, "ChroniX2Grid - The Extensive PowerGrid Time-serie Generator." https://github.com/BDonnot/ChroniX2Grid, 2020.

[19] F. Pourahmadi, H. Heidarabadi, S. H. Hosseini, and P. Dehghanian, "Dynamic uncertainty set characterization for bulk power grid flexibility assessment," *IEEE Systems Journal*, 2019.

[20] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, 2000.

[21] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap *et al.*, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016.

[22] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, ser. Springer series in statistics. Springer, 2009.

[23] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*. Association for Computing Machinery, 2009.

[24] Y. Wu and Y. Tian, "Training agent for first-person shooter game with actor-critic curriculum learning," in *International Conference on Learning Representations*, 2017.

[25] V. Ajjarapu and C. Christy, "The continuation power flow: a tool for steady state voltage stability analysis," *IEEE Transactions on Power Systems*, vol. 7, no. 1, pp. 416–423, 1992.

[26] E. L. Allgower and K. Georg, "The basic principles of continuation methods," in *Numerical Continuation Methods*. Springer, 1990, pp. 7–16.

[27] H.-D. Chiang, T.-Q. Zhao, J.-J. Deng, and K. Koyanagi, "Homotopy-enhanced power flow methods for general distribution networks with

distributed generators," *IEEE Transactions on Power Systems*, vol. 29, no. 1, pp. 93–100, 2014.
[28] P. Crucitti, V. Latora, and M. Marchiori, "Model for cascading failures in complex networks," *Physical Review E*, vol. 69, no. 4, p. 045104, 2004.
[29] L. D. Valdez, L. Shekhtman, C. E. La Rocca, X. Zhang, S. V. Buldyrev, P. A. Trunfio, L. A. Braunstein, and S. Havlin, "Cascading failures in complex networks," *Journal of Complex Networks*, vol. 8, no. 2, 2020.
[30] B. A. Carreras, V. E. Lynch, I. Dobson, and D. E. Newman, "Critical points and transitions in an electric power transmission model for cascading failure blackouts," *Chaos: An interdisciplinary journal of nonlinear science*, vol. 12, no. 4, pp. 985–994, 2002.
[31] A. E. Motter and Y.-C. Lai, "Cascade-based attacks on complex networks," *Physical Review E*, vol. 66, no. 6, p. 065102, 2002.
[32] S. Li, T. Ding, W. Jia, C. Huang, J. P. Catalao, and F. F. Li, "A machine learning-based vulnerability analysis for cascading failures of integrated power-gas systems," *IEEE Transactions on Power Systems*, 2021.
[33] J. Zhao, D. Li, H. Sanhedrai, R. Cohen, and S. Havlin, "Spatio-temporal propagation of cascading overload failures in spatially embedded networks," *Nature communications*, vol. 7, no. 1, pp. 1–6, 2016.
[34] D. P. Nedic, I. Dobson, D. S. Kirschen, B. A. Carreras, and V. E. Lynch, "Criticality in a cascading failure blackout model," *International Journal of Electrical Power & Energy Systems*, vol. 28, no. 9, pp. 627–633, 2006.
[35] L. Zhang, G. Zeng, D. Li, H.-J. Huang, H. E. Stanley, and S. Havlin, "Scale-free resilience of real traffic jams," *Proceedings of the National Academy of Sciences*, vol. 116, no. 18, pp. 8673–8678, 2019.
[36] Z. Yan and Y. Xu, "L2RPN WCCI 2020 Competition - Third Place Solution." https://github.com/ZM-Learn/L2RPN_WCCI_a_Solution, 2020.
[37] Z. Ding, T. Yu, Y. Huang, H. Zhang, L. Mai, and H. Dong, "Rlzoo: A comprehensive and adaptive reinforcement learning library," *arXiv preprint arXiv:2009.08644*, 2020.

APPENDIX

Consider the 3-bus system shown in figure 17. The total load is split between bus-1 and bus-2, with the power flow on the line-1 (line-2) equal to the load on bus-1 (bus-2). The fraction β, that divides the total load between bus-1 and bus-2, is determined by an RL agent based on the total load. The flow limit of line-1 and line-2 is $P_{1,max}$ and $P_{2,max}$ respectively. The total load demand, P_L, is a function of the time-step, n, and is given by (11). Let the RL agent be represented by a sigmoid layer with a single learnable parameter, θ_1. The fraction, β, in terms of the total load and θ_1 is given by (12). The expression (12) implies that β continuously increases with θ_1 and $0 \leq \beta \leq 1$.

$$P_L(n) = P_0 - 0.5 \cdot cos(2\pi n/100) \tag{11}$$

$$\beta(P_L(n), \theta_1) = 1/(1 + e^{-(0.5P_L(n)+\theta_1)}) \tag{12}$$

The goal of the RL agent is to prevent cascading due to line overloads. In this simple example, a line disconnection transfers all its load to the other line in the next time step. This behavior emulates the network flow reconfiguration in meshed power networks after a line outage. The RL-agent needs to be learnt so that the system operates for 100 time-steps without cascading. The learnable parameter, θ_1, of the RL-agent is the value that maximizes the objective function (13), which is the sum of the rewards for each line. When an early termination occurs due to cascading, the rewards are only summed till the termination time-step. The reward function $R(x)$ is given by (16), where α is the overload penalty.

$$J(\theta_1) = \Sigma_n R(P_1(n, \theta_1)/P_{1,max}) + \Sigma_n R(P_2(n, \theta_1)/P_{2,max}) \tag{13}$$

$$P_1(n, \theta_1) = \beta(P_L(n), \theta_1) \cdot P_L(n) \tag{14}$$

$$P_2(n, \theta_1) = (1 - \beta(P_L(n), \theta_1)) \cdot P_L(n) \tag{15}$$

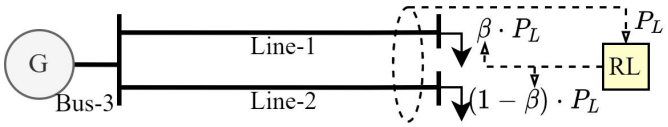


Fig. 17: Simple 3-bus system with RL-Agent controlling β.

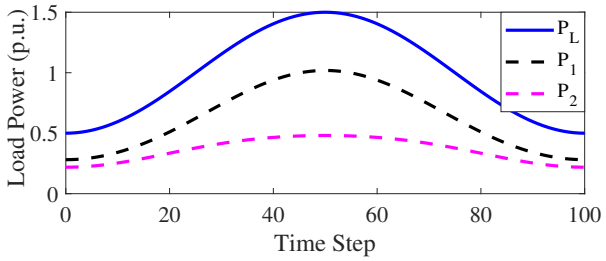


Fig. 18: Variation of P_L, P_1 & P_2 with time step for $\theta_1 = 0$.

$$R(x) = \begin{cases} (0.95 - x), & x \leq 0.95 \\ \alpha \cdot (0.95 - x), & x > 0.95 \\ -100, & \text{early termination} \end{cases} \tag{16}$$

For illustration purposes, we will fix $P_0 = 1.0$, $P_{1,max} = 1.0$ & $P_{2,max} = 0.75$ for the rest of the analysis. Figure 18 plots the variation of P_L, P_1 & P_2 versus the time-step for $\theta_1 = 0$. The three powers are in phase, i.e., they all attain the maximum and minimum at the same time-step. Further, the flows are monotonically increasing for $n < 50$ and monotonically decreasing for $n > 50$ with the peak at $n = 50$. Finally, there is a symmetry arising from cos(x) before and after $n = 50$, with the same flows at a time-step k and the time-step $(50 - k)$. The training of the RL agent is equivalent to finding the optimal value of θ_1 that maximizes the objective function $J(\theta_1)$. The ease of training the RL agent is directly related to the likelihood of estimating the optimal value of θ_1 using stochastic gradient methods. A high likelihood of reaching the optimal value implies faster training, while a low likelihood implies longer training as more simulations of the environment need to be performed before a consistent rise in the objective function is observed.

We will demonstrate how the curriculum approach will aid in the estimation of the optimal θ_1. Three levels of curriculum are considered, and their settings are identical to Table I with the following two modifications - (a) α for level-1 = 2 (b) the SOT for level-2 is 1.2. Level-1 and level-2 are the relaxed versions of the environment while level-3 is the true behavior of the environment. The optimal value of θ_1 for level-x is denoted by θ_1^{x*}.

A. Analysis of Region of Convergence for the Three Levels

For level-1, the SOT and HOT are high (10^9) and so, early termination does not occur for any value of θ_1. Thus, the reward function $R(x)$ for level-1 will be piece-wise linear and concave (as $\alpha = 2$). Further, as β increases continuously when θ_1 is increased, the overall objective $J(\theta_1)$ is also continuous without any sudden transitions. When $\theta_1 << 0 \Rightarrow \beta \approx 0$, most of the power flows through line-2. Hence, when $\theta_1 << 0$, the reward of line-2 becomes negative for many time-steps and the value of the overall objective function is reduced. As θ_1 increases to 0, the overload in line-2 reduces and so the

This article has been accepted for publication in IEEE Transactions on Power Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPWRS.2022.3213487

objective increases. When $\theta_1 \approx 0$, the $P_1 \approx 1$ and $P_2 \approx 0.5$ at the peak power ($n = 50$), which causes much lesser negative rewards. Finally, when $\theta_1 >> 0 \Rightarrow \beta \approx 1$, the reward of line-1 is negative for many time-steps which again reduces the value of the objective function. Thus, the objective function is a continuous concave function that increases to a peak near $\theta_1 \approx 0$. So, the region of convergence using gradient methods for level-1, $ROC_{level-1}$, is [-1.5,1.5] and $\theta_1^{1*} \approx 0$.

For level-2, there is a chance of early termination if $P_1 > 1.2 \cdot P_{1,max}$ or $P_2 > 1.2 \cdot P_{2,max}$ for more than COL (10 time steps). By the symmetry of $P_L(n)$ and the monotonic property of $P_L(n)$ for $n < 50$, the SOT limit of 1.2 must be reached for line-1 or line-2 at $n = 45$ if a disconnection must occur. If this limit is not reached by $n = 45$, then the SOT will not be violated after $n = 55$ and the line will not get disconnected as the COL limit of 10 steps will not be reached. As we discussed in the analysis of level-1, $\theta_1 >> 0$ ($\theta_1 << 0$) implies that the line-1 (line-2) limit gets violated. The values of θ_1 which cause line-1 limit to be violated for more than 10 time-steps satisfy the relations (17)-(19), which are simplified using (14) for $P_1(45, \theta_1)$ and $P_{1,max} = 1$.

$$P_1(45, \theta_1) > 1.2 \cdot P_{1,max} \tag{17}$$

$$\Rightarrow P_L(45)/(1 + e^{-(0.5 \cdot P_L(45) + \theta_1)}) > 1.2 \tag{18}$$

$$\Rightarrow \theta_1 > -log((P_L(45))/1.2 - 1) - 0.5 \cdot P_L(45) \tag{19}$$

Similarly, the values of θ_1 which cause line-2 limit to be reached for more than 10 time-steps satisfy the relations (20)-(22), which are simplified using expression (15) for $P_2(45, \theta_1)$ and $P_{2,max} = 0.75$.

$$P_2(45, \theta_1) > 1.2 \cdot P_{2,max} \tag{20}$$

$$\Rightarrow (1 - 1/(1 + e^{-(0.5 \cdot P_L(45) + \theta_1)})) > 0.9/P_L(45) \tag{21}$$

$$\Rightarrow \theta_1 < -log(0.9/(P_L(45) - 0.9)) - 0.5 \cdot P_L(45) \tag{22}$$

On calculating the numerical values of θ_1 by using (19) & (22) and $P_0 = 1$, the lower and upper limits for θ_1 are found to be -1.18 & 0.74. Thus, the region of convergence for level-2, $ROC_{level-2}$, is [-1.18,0.74]. The optimal value of θ_1 for level-2, θ_1^{2*}, is approximately the midpoint of the ROC, which implies $\theta_1^{2*} \approx -0.22$.

For level-3, the SOL is 1 and the COL is 3 time-steps. Thus, the θ_1 that causes the line limit to be reached by $n = 48$ determines the limits of the ROC. The final expressions for θ_1 to reach the line limits are presented in (23) & (24). On substituting $P_0 = 1$, the region of convergence for level-3, $ROC_{level-3}$, is found to be [-0.05,-0.75].

$$\Rightarrow \theta_1 > -log((P_L(48)) - 1) - 0.5 \cdot P_L(48) \tag{23}$$

$$\Rightarrow \theta_1 < -log(0.75/(P_L(48) - 0.75)) - 0.5 \cdot P_L(48) \tag{24}$$

The theoretical analysis above is used to prove the three propositions discussed in Section V.

B. Proof of Propositions for the 3-bus System

Proposition-1 relates the number of successful time-steps for the three levels. The number of successful steps of level-1 is always 100, which is the maximum possible number of time-steps. As $\text{COL}_{level-2} > \text{COL}_{level-3}$, lines in level-3 disconnect with shorter time delay than level-2 for a fixed θ_1. Thus, the number of steps for level-1 is always more than level-2 which is always more than level-3. This is precisely the claim of Proposition-1.

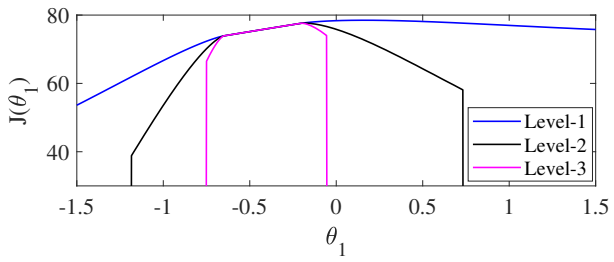


Fig. 19: The objective function versus θ_1 for the three levels.

Proposition-2 relates the ease of training for the three levels. As RL methods essentially use gradient methods, the likelihood of learning is high when θ_1 is initialized in the ROC and the likelihood is low if it is initialized outside the ROC. Thus, the ease of training is directly related to the size of the ROC interval as the probability of initializing in the ROC increases if the ROC interval is large. The ROC of the three levels are related by 25. Hence, level-1 is easier to train than level-2 which is in turn easier to train than level-3. This is precisely the claim of Proposition-2.

$$ROC_{level-3} \subset ROC_{level-2} \subset ROC_{level-1} \tag{25}$$

Proposition-3 describes the transfer learning property of the lower levels to the next level. From the analysis above, we can see that $\theta_1^{1*} \in ROC_{level-2}$ and $\theta_1^{2*} \in ROC_{level-3}$. This implies that θ_1^{x*} from a lower level will perform well on the next level. Furthermore, initializing level-2 training from θ_1^{1*} will lead to a faster convergence than starting from a random θ_1. The same observation holds for level-3 when initializing from θ_1^{2*}. This is exactly the claim of proposition-3 and is the key motivation to use curriculum methods, as initializing from the solution of a simple curriculum will likely lead to faster convergence on a harder curriculum.

To numerically validate the analysis, the 3-bus system is simulated with cascading in MATLAB for the three levels with varying θ_1 between -1.5 and 1.5. The line flows at each step are used to calculate the objective function, $J(\theta_1)$. Figure 19 plots the $J(\theta_1)$ for varying θ_1 for the three levels. it can be seen that the $J(\theta_1)$ curve is smooth for level-1 and no sharp transitions are observed. In contrast, level-2 and level-3 have a sharp cut-off. These boundaries closely match the theoretically estimated ROC intervals for level-2 and level-3, validating the theoretical analysis. Also, figure 19 illustrates that $\theta_1^{1*} \in ROC_{level-2}$ and $\theta_1^{2*} \in ROC_{level-3}$, which numerically validates the transfer learning property between the curricula.

Hence, we have proven analytically and shown numerically the validity of the propositions that ensure that curriculum design will lead to effective RL agent training by relating the region of convergences and θ_1^{x*} between various levels. This is the final contribution of our work.

Amarsagar Reddy Ramapuram Matavalam (M19) received B.Tech. degree in Electrical Engineering and M.Tech. degree in Power Electronics and Power Systems, both from IIT-Madras, Chennai, India ; and a doctorate in electrical engineering from Iowa State University, Ames, IA, USA. He is currently a assistant professor at Arizona State University, Tempe, AZ, USA. His research is in power system stability analysis and control, data-driven approaches for power system analysis, and characterizing uncertainty in power systems.

Kishan Prudhvi Guddanti (S20) received the B.Tech. degree in electrical engineering from Sri Ramaswamy Memorial Institute of Science and Technology, Chennai, India; and M.Sc and Ph.D. degree in electrical engineering from Arizona State University, Tempe, AZ, USA. He is currently a power system engineer at Pacific Northwest National Laboratory, Richland, WA, USA. His current research interest is in the interdisciplinary area of AI applications in power systems in addition to voltage stability, and data-driven techniques for power system risk assessment and control.

Yang Weng (M14) received the B.E. degree in electrical engineering from Huazhong University of Science and Technology, Wuhan, China; the M.Sc. degree in statistics from the University of Illinois at Chicago, Chicago, IL, USA; and the M.Sc. degree in machine learning of computer science and M.E. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University (CMU), Pittsburgh, PA, USA. After finishing his Ph.D., he joined Stanford University, Stanford, CA, USA, as the TomKat Fellow for Sustainable Energy. He is currently an Assistant Professor of electrical, computer and energy engineering at Arizona State University (ASU), Tempe, AZ, USA. His research interest is in the interdisciplinary area of power systems, machine learning, and renewable integration.

Venkataramana Ajjarapu (S'86–M'86–SM'91–F'07) received the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1986. He is currently the Thomas M. Whitney Professor of Electrical Engineering in the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, USA. His research interests include voltage stability analysis, power system security, wind and solar energy integration, real-time control of power and power electronics systems and distributed resources impact on electric grid.