A Deep Reinforcement Learning-based Approach to Post-Disaster Routing of Movable Energy Resources

Mukesh Gautam, *Student Member, IEEE*, Narayan Bhusal, *Member, IEEE*,
Mohammed Benidris, *Senior Member, IEEE*
Department of Electrical & Biomedical Engineering, University of Nevada, Reno
(emails: {mukesh.gautam, bhusalnarayan62}@nevada.unr.edu, mbenidris@unr.edu)

***Abstract*—After the occurrence of an extreme event, movable energy resources (MERs) can be an effective way to restore critical loads to enhance power system resilience when no other forms of energy sources are available. Since the optimal locations of MERs after an extreme event are dependent on system operating states (e.g., loads at each node, on/off status of system branches, etc.), existing analytical and population-based approaches must repeat the entire analysis and computation when the system operating states change. Conversely, deep reinforcement learning (DRL)-based approaches can quickly determine optimal or near-optimal locations despite changes in system states if they are adequately trained with a variety of scenarios. The optimal deployment of MERs to improve power system resilience is proposed using a Deep Q-Learning-based approach. If they are available, MERs can also be used to supplement other types of resources. Following an extreme event, the proposed approach operates in two stages. The distribution network is modeled as a graph in the first stage, and Kruskal's spanning forest search algorithm (KSFSA) is used to reconfigure the network using tie-switches. The optimal or near-optimal locations of MERs are determined in the second stage to maximize critical load recovery. A case study on a 33-node distribution test system demonstrates the effectiveness and efficacy of the proposed approach for post-disaster routing of MERs.**

***Index Terms*—Deep Q Network, distribution system, movable energy resources, reinforcement learning, resilience.**

I. Introduction

Over the last decade, the frequency of extreme events, both natural (e.g., hurricanes, wildfires, ice or hail storms, and earthquakes) and man-made (e.g., cyber and physical attacks), has increased dramatically. For example, there were 20 weather related catastrophic events in the United States in 2021 alone, each with costs surpassing $1 billion [1]. Such extreme events have resulted in damage to important power system equipment resulting in system-wide extended power outages. The electric companies' goal of delivering reliable and resilient electrical supply to its customers has been compromised by catastrophic weather events and subsequent outages. As a result, effective power distribution service restoration (PDSR) procedures must be established in order to reduce the impact of these incidents on end-user customers. PDSR's major goal is to reduce load curtailments and outage duration by making the best use of available resources. Smart grid technologies, such as microgrid formation, network reconfiguration (NR), repair crew dispatch, distributed generation (DG), energy storage, movable energy resources (MERs), and combinations of these methods and techniques, have proven to be the most effective PDSR solutions in this context.

In the literature, several analytical and intelligent search techniques for PDSR based on MERs have been developed to improve the reliability and resilience of the distribution system. A robust optimization framework based on two stages has been developed in [2] for routing and scheduling MERs to enhance the resilience of distribution systems. A two-stage PDSR strategy based on mixed-integer linear programming (MILP) has been proposed in [3] to enhance seismic resilience of distribution systems with MERs. A mixed integer linear programming-based PDSR strategy has been proposed in [4] for an active distribution system, where routing and scheduling of mobile energy storage systems is performed for enhanced resilience. In [5], a two-stage optimization strategy has been proposed to enhance distribution system resilience with mobile energy storage units, where dynamic microgrid formation is also considered. A genetic algorithm-based approach has been developed in [6] to enhance distribution system reliability using MERs. The analytical and population-based intelligent search techniques utilized for PDSR based on MERs to enhance distribution system reliability and resilience have the following shortcomings. The accuracy and efficacy of analytical-based approaches are dependent on the accuracy of the models utilized, with accurate models imposing scalability challenges. Furthermore, mathematical models are typically derived using many approximations and require entire system information. Due to the enormous search space, population-based approaches, on the other hand, are computationally intensive, especially as system sizes increase.

Since learning-driven models can address uncertainty by extracting information from previous data, they have been utilized to overcome the shortcomings of analytical and population-based approaches. Furthermore, because of their capacity to employ information gathered from previous data to solve for new scenarios, learning-driven models do not need to be solved whenever new scenarios are encountered. Reinforcement learning (RL)-based systems are among the learning-driven approaches that can learn from experiences during online operations [7], [8]. Also, RL-based approaches are the best fit for online decision-making applications. Therefore, a learning-based approach for after-event MER dispatch is investigated in this paper for distribution system resilience enhancement.

This paper proposes a deep reinforcement learning (DRL)-based framework for after-event dispatch of MERs to enhance distribution system resilience. The proposed DRL approach is based on the training of a neural network that makes the best decision based on previous experiences [9]. Given a specific decision, the sequential decision process gives a reward value as a function of system outcome. The objective of the proposed DRL agent is to minimize critical load curtailment. To ensure realistic representation of distribution system operations, system constraints including radiality and power balance constraints are considered. In the training phase of the proposed framework, Q values are predicted using forward propagation of a deep neural network (DNN). Actions are selected using the Epsilon-Greedy algorithm. When actions are passed through the training environment, the DRL agent gets rewarded (or penalized) based on its performance. Target Q values are calculated based on the reward. The mean squared error (MSE), which is the most commonly employed loss function for regression, is computed using the predicted and target Q values. Errors are then back-propagated to update the weights of DNN. The trained DRL agent is then used to find the optimal or near-optimal locations for MER deployment. The proposed framework is validated through a case study on a 33-node distribution test system, and the results show that the proposed framework can effectively find an optimal network configuration and MER deployment locations thereby minimizing critical load curtailment.

The remainder of the paper is laid out as follows. The mathematical formulation of the after-event reconfiguration and MER deployment problem is explained in Section II. The proposed framework and solution approach are described in Section III. A case study on the 33-node system is used to validate the proposed work in Section IV. Section V provides some concluding remarks.

II. Mathematical Modeling

This paper combines network reconfiguration (first stage) and MERs routing (second stage) to minimize load curtailments after extreme events. The graph theory-based modeling of the distribution network and the mathematical formulation of the problem under study are presented in this section. In addition, states, actions, and the reward function are described in the context of the problem.

A. Graph Theoretic Modeling of Distribution Network

Distribution systems are equipped with sectionalizing switches (normally closed) and tie-switches (normally open). When all the switches of a distribution network are closed, a meshed network is formed, and the meshed network thus formed can be represented by an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is a set of nodes (or vertices) and $\mathcal{E}$ is a set of edges (or branches). For the MER deployement problem proposed in this paper, the status of tie switches are changed in such a way that radiality is always maintained and microgrids are formed after deployment of MERs.

1) Spanning Tree: A spanning tree is defined as a subset of the undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ that has a minimal number of edges linking all vertices (or nodes). In a spanning tree, the number of edges is one less than the number of vertices. There are no cycles in a spanning tree, and all of the vertices are connected [10]. A linked graph can have many spanning trees, each of which has the same number of edges and vertices. Each of the undirected graph $\mathcal{G}$'s edges has a specific value (or weights). The edge weights vary depending on the problem. The sum total of all edge weights of a spanning tree is minimized when establishing the minimum spanning tree. Fig 1 shows a spanning tree of a hypothetical 12-node system. The spanning tree shown in the figure consists of all system nodes (i.e., 12) and 11 closed branches (edges).

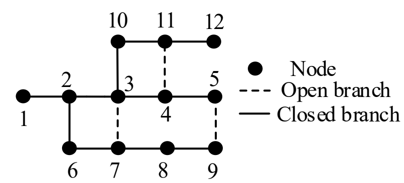

Fig. 1. A spanning tree of a hypothetical 12-node system

2) Spanning Forest: In graph theory, a forest is a disconnected union of trees. A spanning forest is a forest that covers all vertices of the undirected graph $\mathcal{G}$ and consists of a set of disconnected spanning trees [10]. When all spanning trees are connected, each vertex of the the undirected graph $\mathcal{G}$ are included in one of the spanning trees [11]. On the other hand, when a disconnected graph has many connected components, a spanning forest is formed and it contains a spanning tree of each component [12]. Fig. 2 shows the spanning forest formed as a result of disconnection of two addition branches (2–6 and 3–10) in the spanning tree presented in Fig. 1. The spanning forest shown in Fig. 2 consists of three spanning trees (ST-1, ST-2, and ST-3).

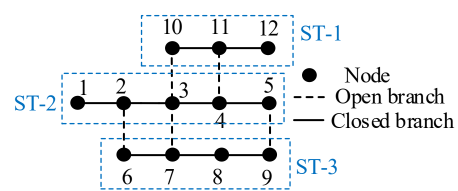

Fig. 2. A spanning forest of the hypothetical 12-node system (in Fig. 1)

In this paper, Kruskal's algorithm [13] is used to search for the optimal spanning forest. The Kruskal's spanning forest search algorithm (KSFSA) starts by constructing a forest F with each graph vertex acting as a single tree based on the given undirected graph. Since KSFSA is a greedy algorithm, it goes on connecting the next least-weight edge that avoids loop or cycle to the forest F at each iteration. The resulting forest F after the last iteration is the optimal spanning forest. Fig. 3 shows the flowchart of KSFSA.

B. Problem Formulation

This subsection presents the objective function and the constraints of the problem under consideration.

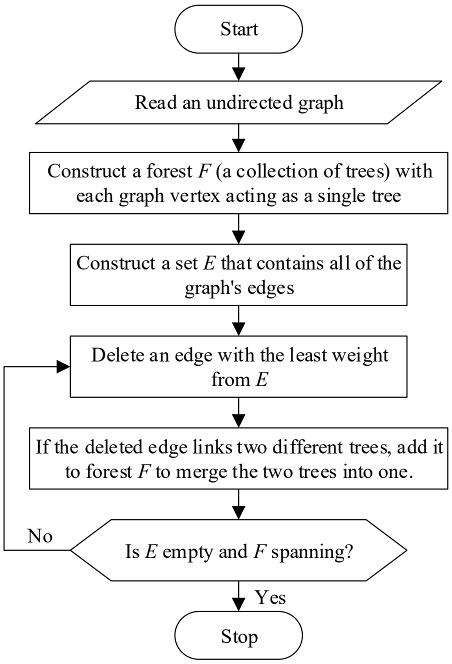

Fig. 3. Flowchart of Kruskal's spanning forest search algorithm

1) Objective Function: As a result of an extreme event, some or all parts of the system may lose power supply. Under such a circumstance, tie-switches should be used to reconfigure the network, and MERs should be deployed to enhance the distribution system's resilience. Therefore, the objective of the after-event MER routing problem under consideration is to minimize the critical load curtailment of the system since it can capture the severity of the multiple line outages and is directly affected by the topology or configuration and MER deployment locations in a distribution system. Mathematically, the objective function of the critical load curtailment minimization is expressed as follows.

$$\text{Min} \sum_{i=1}^{N} \omega_i \Delta P_i, \tag{1}$$

where ΔP_i is the load curtailment at node i; ω_i is the critical load factor at node i; and N is the total number of nodes in the system.

2) Constraints: The problem under consideration is subjected to various constraints including nodal power balance constraints and radiality constraint.

(a) Node power balance constraints: The power balance constraints at each node of the system can be expressed as follows.

$$\sum_{j \in \Omega_g(j)} P_{g,j} + \sum_{l \in \Omega_L(j)} P_{l,j} = P_{D,j} \tag{2}$$

where $\Omega_g(j)$ is the set of sources (including MER) connected to node j; $\Omega_L(j)$ is the set of lines connected to node j; $P_{g,j}$ is the power injected from source j; $P_{D,j}$ is the load at node j; and $P_{l,j}$ is the line power flow from node l to node j.

(b) Radiality constraint: A distribution system must always meet the radiality requirement. Therefore, each potential configuration should be radial (i.e., the radiality constraint should be met for each spanning tree of the network). Each spanning tree of the network is represented by a sub-graph $\mathcal{G}_s = (\mathcal{N}_s, \mathcal{E}_s)$, where $\mathcal{N}_s$ is a set of nodes (or vertices) and $\mathcal{E}_s$ is a set of edges (or branches) in the sub-graph. For the sub-graph, a node-branch incidence matrix should be constructed. If $n = |\mathcal{N}_s|$ denotes the number of nodes and $e = |\mathcal{E}_s|$ denotes the number of edges of a particular spanning tree, then the node-branch incidence matrix $A \in \mathbb{R}^{n \times e}$ is the matrix with element a_{ij} calculated as follows [14].

$$a_{ij} = \begin{cases} +1 & \text{if branch } j \text{ starts at node } i \\ -1 & \text{if branch } j \text{ ends at node } i \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

If the node-branch incidence matrix A is full ranked, then the radiality constraint is satisfied.

C. States, Actions, and Reward Function

The choice of states, actions, and the reward function plays a critical role for the proper training of a reinforcement learning (RL) agent. States, actions, and reward function must be, therefore, chosen with careful consideration. For the MER routing problem under consideration, a vector of on/off status of the network edges after reconfiguration is taken as the state. The action is a vector of MER deployment locations. The cost function at time step t is expressed as follows.

$$C_t = LC_t^{RL} - LC_t^{min}, \tag{4}$$

where LC_t^{RL} is the critical load curtailment at time step t as a result of the action taken by the RL agent; and LC_t^{min} is the minimum critical load curtailment at time step t.

The total reward at time step t is computed as follows.

$$R_t = \begin{cases} \frac{100}{1+C_t} & \text{if all constraints are satisfied} \\ -\rho & \text{if any constraint is violated} \end{cases} \tag{5}$$

where ρ denotes the penalty factor.

III. Reinforcement Learning for MER Routing

This work leverages recently advanced reinforcement learning techniques for after-event MER routing to minimize the critical load curtailment following an extreme event. This section provides a brief overview of Deep Q learning and its training attributes.

A. Deep Q Learning

The four main integrands of a reinforcement learning (RL)-based system are policy, reward, value functions, and the environment model. An agent decides what action to take based on the policy. The policy establishes a relationship between states and actions. When the agent performs a task, it is rewarded (or penalized). The value function determines the expected value of cumulative reward when an agent follows a policy. There are a variety of algorithms for RL. A number of factors influence the choice of an algorithm, such as the nature of states (continuous or discrete), the action-space (continuous or discrete), and so on. The action-space for the MER routing problem under consideration is discrete, making Q-Learning an appropriate option for the task. Basic Q-Learning, on the other hand, necessitates large look-up tables to store state-action values. As an action-value function approximator, a deep neural network (DNN) is employed to avoid the usage of large look-up tables. The addition of DNN to the basic Q-Learning framework transforms it into a Deep Q Network (DQN). The update rule for action-value function in Q-learning is defined as follows [7].

$$\begin{aligned} Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \times [R_{t+1} \\ +\gamma \times \max_a Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)] \end{aligned} \tag{6}$$

where A_t and S_t are the action and state of an agent at t^{th} iteration; $Q(S_t, A_t)$ is the action-value function at t^{th} iteration; $Q(S_{t+1}, A_{t+1})$ is the action-value function at $(t+1)^{\text{th}}$ iteration; α is the learning rate; and γ is the reward discount factor.

Instead of updating the action-value function iteratively, the DNN is trained and the action-value function's parameters are optimized to minimize the mean-squared error (MSE) loss function (i.e., regression loss function), which is expressed as follows [15].

$$L(\theta) = \mathbb{E}[(Q(S_t, A_t|\theta) - y_t)^2], \tag{7}$$

where $\mathbb{E}$ denotes expectation operator; θ denotes the parameter of action-value function $Q(S_t, A_t)$; and y_t denotes the target action-value function, which is defined as follows.

$$y_t = R(S_t, A_t) - \gamma \times Q(S_t, A_t; \phi). \tag{8}$$

In (8), $R(S_t, A_t)$ denotes the reward function at the t^{th} iteration; ϕ denotes the parameter of the target DQN; and $Q(S_t, A_t; \phi)$ denotes the action-value function of the target DQN.

B. Training Attributes

The experience replay memory-based training of DQN is performed for a certain number of episodes (n_{ep}). The parameters θ of the main DQN are initialized with some random values and the parameters ϕ of the target DQN are set equal to that of the main DQN. Each episode starts by initializing the system with a random state, which is a vector of on/off status of the network branches (or edges) after reconfiguration. In each time step, the predicted Q values corresponding to each action is computed based on forward propagation of DNN. For the selection of actions, the Epsilon-Greedy (exploration-exploitation) algorithm [16] is used. The value of exploration rate, ε, is initialized at 1. The epsilon is updated after each episode as follows.

$$\varepsilon_{new} = \varepsilon_{old} - \frac{\varepsilon_{old} - \varepsilon_{min}}{0.25 \times n_{ep}}, \tag{9}$$

where ε_{min} is the minimum exploration rate. The target Q value of the DQN is computed using (8). The experience replay memory is appended with transition $(S_t, A_t, R_{t+1}(S_t, A_t))$. MSE losses for each time step t are computed based on (7) using the predicted Q-value of the main DQN and target Q-values. The parameters of the main DQN are updated by back-propagating these MSE losses. After a certain number of iterations, the parameters of the target DQN are periodically updated.

Algorithm 1 provides the procedure of training the proposed DRL-based MER routing problem.

Algorithm 1: Training of the proposed DRL-based MER routing problem

```
Input   : System data including line data, load data,
          on/off status of branches, etc.
Initialize experience replay memory M
Initialize parameters θ of main DQN with random
  values
Set target DQN parameters φ equal to main DQN
  parameters, i.e., φ ← θ
for episode ← 1 to n_ep do
    Initialize the system with a random state (here, a
      vector of line/branch status)
    for t ← 1 to T do
        Generate action-value function Q based on
          current state
        Calculate the reward function R_{t+1}(S_t, A_t)
          after passing the state and action-value
          function through reward generator
        Append the experience replay memory M with
          transition (S_t, A_t, R_{t+1}(S_t, A_t))
        if length(M) > batch_size then
            Randomly select a minibatch
            Calculate DQN Loss Function based on
              main Q-function and target Q-function
            Perform back-propagation to update
              parameters θ of main DQN
            Periodically update parameters φ of target
              DQN
Output : MER deployment locations
```

IV. Case Study and Discussion

A. System Description

To demonstrate the effectiveness of the proposed approach, the 33-node system is used for numerical simulations. The 33-node distribution test system is a radial distribution system with 33 nodes, 32 branches, and 5 tie-lines (37 branches) [17]. As shown in Fig. 4, all branches (including tie-lines) are numbered from 1 to 37. The system's overall load is 3.71 MW.

The locations and amounts of critical loads considered for the 33-node system are shown in Table I. The hyper-parameter settings of the main and target DQNs of the proposed framework for the 33-node system are shown in Table II.

TABLE I
LOCATIONS OF CRITICAL LOADS FOR THE 33-NODE SYSTEM

Nodes	4	5	6	7	8	9	10	11	18	19	20	21	22	23	26	27	28	29	30	33
Critical Loads (kW)	60	30	60	200	200	60	30	25	45	45	45	45	45	45	60	60	60	60	60	30

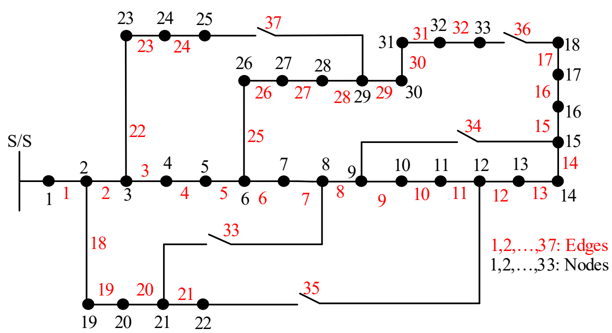

Fig. 4. 33-node distribution test system

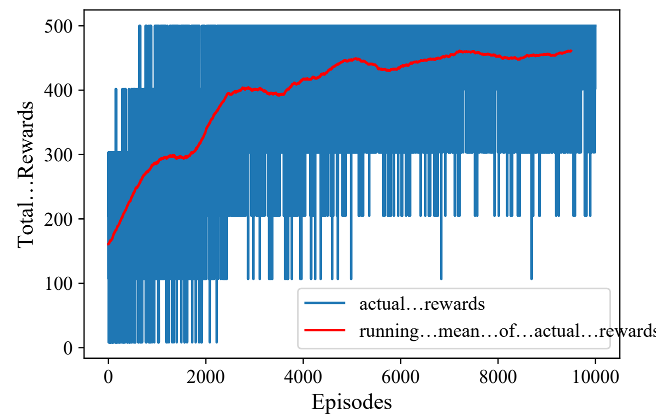

Fig. 5. Total rewards of training episodes

TABLE II
HYPER-PARAMETER SETTINGS OF MAIN AND TARGET DQNS FOR 33-NODE SYSTEM

Hyper-parameters	Values
Number of hidden layers	3
No. of neurons in hidden layers	10, 10, 10
Learning rate	10^{-3}
Reward discount factor	0.99
Activation function of output layer	Linear
Activation function of hidden layers	ReLU
Optimizer	Adam
Replay memory size	10000
Batch size	200
Target DQN parameters update rate	50 iterations

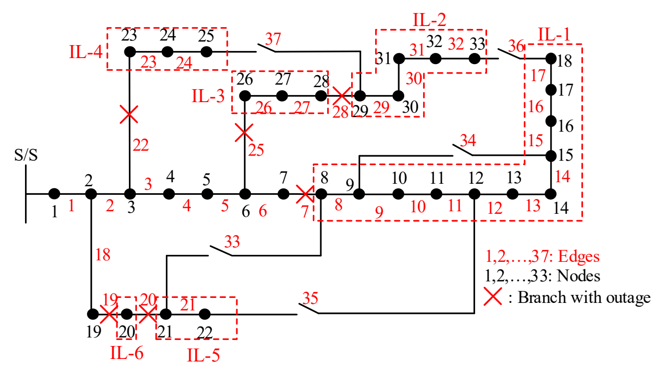

Fig. 6. Test Case-I before reconfiguration and MER deployment

B. Training

The training of the proposed framework for the 33-node system is performed for 10000 episodes. The parameters θ of the main DQN are initialized with random values and the parameters ϕ of the target network are set equal to θ. In each episode, the system is initialized with a random state and action-value function is generated based on the current state. The reward function is calculated by passing the state and action-value function through reward generator. Initially, the rewards are very low but then they increase as the number of episodes increases. Fig. 5 shows the running mean (500-episode window) of actual rewards as the episode progresses. It can be seen from the figure that as the number of episodes increases, the running mean of the reward increases and almost saturates after nearly 7000 episodes.

C. Testing and Implementation

For the testing and implementation of the trained model, two test cases are devised with different line outage scenarios. The two test cases are explained below.

1) Test Case-I: In this case, the outage of six lines 7, 19, 20, 22, 25, and 28 are simulated. Due to the outage of these lines, six isolates (IL-1, IL-2, IL-3, IL-4, IL-5, and IL-6) are formed as shown in Fig. 6. These isolates are devoid of power supply. This results in the total critical load curtailment of 870 kW.

When the outage data are given as inputs to the proposed DRL model, three tie-switches (33, 36, and 37) are closed and MERs are placed at nodes 4, 8, 27, and 30. This results in the formation of two microgrids (MG-1 and MG-2) and an isolate (IL-1), as shown in Fig. 7. In MG-1, the total generation is 600 kW but the total critical load is 645 kW; this results in the critical load curtailment of 45 kW. In MG-2, the total generation exceeds the total critical load, resulting in no critical load curtailment. The total critical load in IL-1 is 45 kW. Therefore, the total critical load curtailment after reconfiguration and MER deployment is 90 kW. The proposed approach is able to recover 780 kW of critical loads for the given outage scenario.

2) Test Case-II: In this case, the proposed approach is tested with a more extreme outage scenario, where outage of the line connected to the substation node (i.e., 1) is considered in addition to outage of lines 6, 10, 17, 19, and 25, as shown in Fig. 8. Because of the outage of the line connected to the substation node, this results in the power interruption at all system nodes and the total critical load curtailment in this scenario is 1265 kW.

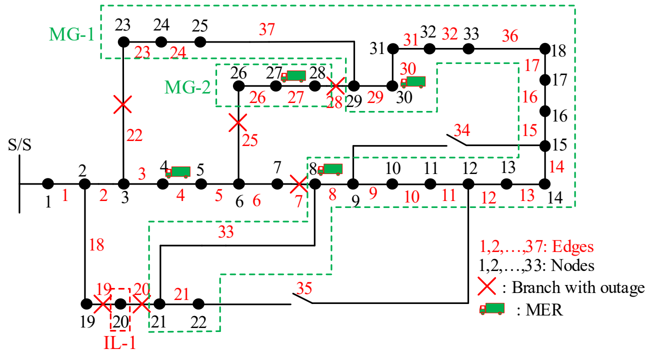

Fig. 7. Test Case-I after reconfiguration and MER deployment

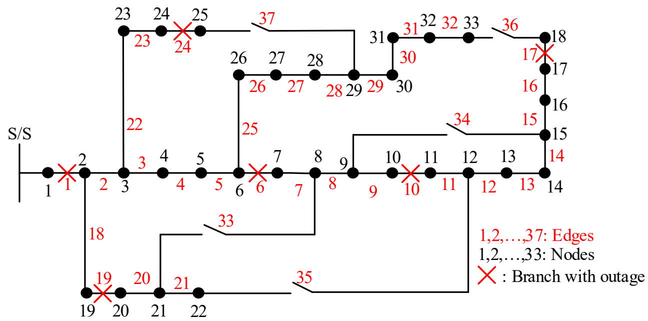

Fig. 8. Test Case-II before reconfiguration and MER deployment

After the implementation of the proposed DRL approach for this test case, four tie-switches 33, 34, 36, and 37 are closed and MERs are placed at nodes 8, 20, 27, and 30. Two microgrids (MG-1 and MG-2) are formed without any isolates, as shown in Fig. 9. In MG-1, the total critical load is 615 kW and the total generation is 600 kW, resulting in the critical load curtailment of 15 kW. Similarly, in MG-2, the total critical load is 650 kW and the total generation is 600 kW, resulting in the critical load curtailment of 50 kW. Therefore, the total critical load curtailment is 65 kW. The total critical load recovered by the proposed approach is 1200 kW for the given scenario.

The proposed approach takes approximately 4 milliseconds to execute on a PC with a 64-bit Intel i5 core processor running at 3.15 GHz, 8 GB RAM, and Windows OS.

V. Conclusion

This paper has proposed a DRL-based two-stage approach for network reconfiguration and MER routing to minimize

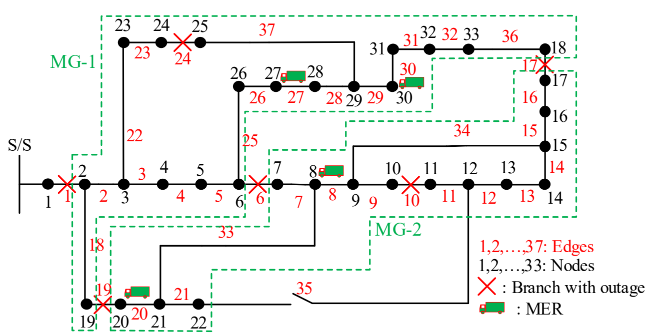

Fig. 9. Test Case-II after reconfiguration and MER deployment

critical load curtailment when multiple line outages occur following an extreme event. In the first stage, distribution network reconfiguration is performed using tie-switches. In the second stage, MERs are utilized to form microgrids. The distribution network was represented by an undirected graph and the optimal spanning forest was formed. The proposed approach was tested and implemented on the 33-node distribution test system. The two test cases exhibit the effectiveness of the proposed approach for recovering critical loads of the system by utilizing MERs and forming microgrids.

Acknowledgement

This work was supported by the U.S. National Science Foundation (NSF) under Grant NSF 1847578.

References

[1] NOAA National Centers for Environmental Information (NCEI), “U.S. billion-dollar weather and climate disasters,” 2022.

[2] S. Lei, C. Chen, H. Zhou, and Y. Hou, “Routing and scheduling of mobile power sources for distribution system resilience enhancement,” *IEEE Transactions on Smart Grid*, vol. 10, no. 5, pp. 5650–5662, 2018.

[3] Z. Yang, P. Dehghanian, and M. Nazemi, “Enhancing seismic resilience of electric power distribution systems with mobile power sources,” in *2019 IEEE Industry Applications Society Annual Meeting*. IEEE, 2019, pp. 1–7.

[4] M. Nazemi, P. Dehghanian, X. Lu, and C. Chen, “Uncertainty-aware deployment of mobile energy storage systems for distribution grid resilience,” *IEEE Transactions on Smart Grid*, vol. 12, no. 4, pp. 3200–3214, 2021.

[5] J. Kim and Y. Dvorkin, “Enhancing distribution resilience with mobile energy storage: A progressive hedging approach,” in *2018 IEEE Power & Energy Society General Meeting (PESGM)*. IEEE, 2018, pp. 1–5.

[6] F. M. Rodrigues, L. R. Araujo, and D. R. Penido, “A method to improve distribution system reliability using available mobile generators,” *IEEE Systems Journal*, vol. 15, no. 3, pp. 4635–4643, 2020.

[7] A. Zai and B. Brown, *Deep reinforcement learning in action*. Manning Publications, 2020.

[8] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.

[9] M. Gautam, N. Bhusal, and M. Benidris, “Deep Q-Learning-based distribution network reconfiguration for reliability improvement,” in *2022 IEEE/PES Transmission and Distribution Conference and Exposition (T&D)*. IEEE, 2022, pp. 1–5.

[10] A. Gibbons, *Algorithmic graph theory*. Cambridge university press, 1985.

[11] A. Salles da Cunha, L. Simonetti, and A. Lucena, “Formulations and branch-and-cut algorithm for the k-rooted mini-max spanning forest problem,” in *International Conference on Network Optimization*. Springer, 2011, pp. 43–50.

[12] I. K. Ahani, M. Salari, S. M. Hosseini, and M. Iori, “Solution of minimum spanning forest problems with reliability constraints,” *Computers & Industrial Engineering*, vol. 142, p. 106365, 2020.

[13] A. Katsigiannis, N. Anastopoulos, K. Nikas, and N. Koziris, “An approach to parallelize kruskal’s algorithm using helper threads,” in *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*. IEEE, 2012, pp. 1601–1610.

[14] M. Gautam and M. Benidris, “Distribution network reconfiguration using deep reinforcement learning,” in *2022 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*. IEEE, 2022, pp. 1–6.

[15] D. Cao, W. Hu, J. Zhao, G. Zhang, B. Zhang, Z. Liu, Z. Chen, and F. Blaabjerg, “Reinforcement learning and its applications in modern power and energy systems: A review,” *Journal of Modern Power Systems and Clean Energy*, vol. 8, no. 6, pp. 1029–1042, 2020.

[16] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[17] M. E. Baran and F. F. Wu, “Network reconfiguration in distribution systems for loss reduction and load balancing,” *IEEE Power Engineering Review*, vol. 9, no. 4, pp. 101–102, 1989.