LARA: LLM-based Agile Power Distribution Network Restoration from Disastrous Events

Anonymous ACL submission

Abstract

Restoring power distribution networks after disruptions demands rapid, reliable coordination across repair crews, mobile power sources, and switching actions under strict constraints. Classical optimization yields high-quality plans but can be slow, while reinforcement learning often requires feeder-specific training and careful reward shaping. We recast restoration as language-conditioned planning: a large language model generates high-level restoration plans over a compact pre-validated catalogue of feasible actions. This constrained generation design makes decisions reliably, scalably, and interpretably, and allows for real-time human-in-the-loop decision-making while requiring no topology-specific setup or retraining. Our method achieves near-mixed-integer-linear programming performance on the IEEE 13-node standard power distribution feeder and outperforms a time-capped MILP solver on the IEEE 33-node standard feeder by around 13%, while using less than 1% of its wall-clock runtime.

1 Introduction

The timely restoration of power distribution networks after extreme weather events and natural disasters is critical to the energy resilience of modern societies (Shen et al., 2020). It is a complex sequential decision-making problem, requiring optimally coordinated actions including dispatching repair crews (RCs), routing and connecting mobile power sources (MPSs), and executing radiality-safe switch operations under tight resource and operational constraints. These actions determine which power loads are energized at each time step, and thus directly impact the scale and duration of power outages after disastrous events. In practice, multiple factors complicate the decision-making, e.g., RCs travel and repair over multiple time steps, MPSs move and supply limited power to isolated islands of the network, and only loop-free switch closures are legal; restoration is therefore a temporally coupled, constraint-laden combinatorial decision-making problem rather than a single-shot topology tweak (benchmarked on standard IEEE 13/33-node feeders) (Kersting, 2001; Baran and Wu, 1989).

A large body of work frames restoration as mixed integer linear programming (MILP) optimization that jointly schedules repair-crew routing/repair, MPS placement/dispatch, and feeder reconfiguration under radiality and power limits (time-indexed over a horizon) (Lei, 2019; Lei et al., 2019; Ye et al., 2021; Ding et al., 2020; Yang et al., 2023; Hom, 2022; Wan, 2024; Kim and Dvorkin, 2019; Nazemi et al., 2021; Yao et al., 2020). These models deliver high-quality plans but are computationally intensive, so practical real-time implementation during disaster response relies on time caps or horizon truncation. In our experiments we use the MILP formulation of Lei (2019) as an oracle upper bound of small systems; for larger systems we implement MILP with time caps Ding et al. (2020) as a meaningful baseline method.

To avoid repeated large-scale optimization problem solutions, recent work casts restoration as a cooperative partially observable Markov game and applies centralized-training, decentralized-execution (CTDE) multi-agent reinforcement learning (MARL). Popular choices include proximal policy optimization (PPO) based algorithm with a centralized critic for coordination (Sch, 2017; Yu2, 2021), and value-factorization methods such as Value Decomposition Networks and Q-Mixing Networks that decompose the joint action-value while enabling decentralized execution (Sun, 2018; Ras, 2018). Applied to restoration problems, CTDE agents coordinate RC/MPS actions under action masking and reward shaping, showing fast inference at test time but requiring substantial training steps and careful reward shaping, with transfer brittleness across feeders/fault patterns (Jacob et al., 2024; Wang et al., 2024). Hierarchical MARL mitigates long-horizon credit assignment by introduc-

ing macro-choices, yet still lacks hard feasibility guarantees without explicit guards.

The complementary limits of MILP (slow, topology-dependent runtimes) and RL (training- and topology-specific policies with brittle transfer) motivate a third path: *language-conditioned planning over constrained affordances*. Large language models (LLMs) can propose joint actions in a fraction of the time, require no topology-specific tuning, accept mid-episode human updates, and produce plans that are interpretable. Prior work in language-conditioned control shows how affordances/tools curb free-form generation and offload arithmetic and symbolic checks (Ahn et al., 2022; Yao et al., 2023; Schick et al., 2023); diversity-and-selection further improves reliability under tight budgets (Vijayakumar et al., 2016; Freitag et al., 2022). To our knowledge, no prior LLM system addresses this restoration problem with joint RC-MPS-switch coordination and explicit safety guarantees. In this paper, we propose *LLM-based Agile Power Distribution Network Restoration from DisAstrous Events* (LARA), which follows this affordance-first, language-conditioned planning paradigm. Our contributions are:

- We formulate power distribution network restoration as a *language-conditioned planning* problem, enabling LLMs to reason over constrained, prevalidated action spaces.
- LARA proposes an affordance-first, language-driven planning approach that decouples feasibility computation from LLM reasoning.
- LARA achieves near–MILP performance and exceeds time-capped MILP and RL baselines by close to 13% on IEEE 13/33-node feeders with zero constraint violations using less than 1% of the wall-clock runtime of the MILP.

2 Methodology

In this section, we detail the LARA method and begin by formulating the problem in Section 2.1. Figure 1 provides a high-level workflow: (1) greedy switch reconfiguration handled by the environment (Sec. 2.2), (2) affordance precomputation exposing only topology-safe options (Sec. 2.3), (3) LLM synthesis of RC/MPS macro-actions over those IDs (Sec. 2.5), and (4) validate–execute–replan until any agent becomes idle is executed by the simulation environment); we also support brief human edits mid-episode (Sec. 2.6).

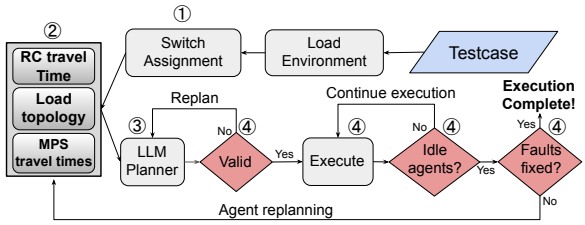

Figure 1: LARA architecture diagram

2.1 Problem Formulation

We model post-disaster restoration as a discrete-time process $t = 0, \ldots, T$ on a radial feeder with a single source and many load nodes. Each load n has active/reactive demand (P_n, Q_n) and a priority weight $W_n \in \{1, \ldots, 10\}$. Each damage point (DP) k lies on a line and has remaining repair workload w_k. The controllable agents are RCs, MPSs, and switches. Each RC has a travel speed and a repair rate (work units per step); each MPS has a travel speed, maximum real/apparent power $(P_{\max}, S_{\max})$, and a finite energy budget. An *island* is a group of nodes that currently cannot receive power from the source because faults or open switches break the path. Let S denote the set of controllable switches and $S_{\text{open},t} \subseteq S$ the subset that is open at time t.

At time t, the state s_t includes agent positions/resources, the open/closed switch set, $\{w_k\}$, and the energized loads L_t. A joint macro-action is

$$a_t = \big(\alpha_{\text{crew},t},\ \alpha_{\text{MPS},t},\ \alpha_{\text{switch},t}\big),$$

where $\alpha_{\text{crew},t}$ assigns each crew to a fault (or idle), $\alpha_{\text{MPS},t} = (m_t, v_t, C_t)$ selects an MPS m_t, a connection node v_t, and a feasible subset of loads C_t to supply, and $\alpha_{\text{switch},t} \subseteq S_{\text{open},t}$ is a set of switch closures. The simulator advances the state via $s_{t+1} = F(s_t, a_t)$, where F is a simulator transition function that updates the state, by applying agent travel, repair progress, switch toggles and MPS connection.

We maximize cumulative prioritized load restored:

$$\max_{\{a_t\}_{t=0}^{T}} \sum_{t=0}^{T} \sum_{n \in L_t} P_n W_n. \tag{1}$$

Hard constraints (enforced by the simulator). (i) *Radiality*: only legal (cycle-free) switch closures are allowed each step; the feeder remains loop-free at all times. (ii) *MPS power limits*: whenever an MPS supplies C_t, the real and apparent power satisfy $\sum_{n \in C_t} P_n \leq P_{\max}$ and

$\sqrt{\left(\sum_{n\in C_t} P_n\right)^2 + \left(\sum_{n\in C_t} Q_n\right)^2} \leq S_{\max}$. (iii) *MPS energy budget*: each MPS has finite energy and cannot supply once depleted. (iv) *Kinematics/timing*: RCs and MPSs move at finite speeds; repairing fault k by a crew with repair rate r takes time w_k/r; a switch closure consumes one step; an MPS requires travel plus a one-step connection before supplying. All feasibility checks are applied before actions execute, and F respects these dynamics when producing s_{t+1}.

2.2 Greedy Switch Reconfiguration

Switch selection is performed by the simulator using a fast greedy procedure that avoids the $2^{|\mathcal{S}|}$ combinatorial search an LLM cannot reliably execute. A reconfiguration is triggered whenever a reconfiguration is required prior to the LLM call. Starting from the current open/closed set, we enforce legality by preserving radiality via a reachability/loop check. For each still-open legal switch s, we compute its marginal benefit $\Delta(s)$, the newly energized weighted load if s were closed using a Breadth-first search from source node with island accounting, which is $O(|\mathcal{N}|)$. We then close the switch with the largest $\Delta(s) > 0$, update the graph, recompute all $\Delta(\cdot)$ values, and repeat until no positive-gain switch remains. This yields $O(|\mathcal{S}| \cdot |\mathcal{N}|)$ work per reconfiguration event which remains a modest computational cost even on large feeders.

2.3 Precomputation

From the current state we build compact tables (IDs with vetted numbers): (i) RC to DP: for each crew-fault pair, travel time, single/paired completion times (T_1, T_2), and a time-weighted benefit proxy; (ii) MPS to Island: for each island, nearest valid node and feasible load-combinations that satisfy $P_{\max}$ and $S_{\max}$, each with travel/connect time and benefit. All feasibility checks (radiality, MPS limits, DP capacity) are enforced here.

2.4 LLM Plan Synthesis (RC/MPS Only)

The model receives the RC to DP and MPS to Island tables and returns a structured plan with`crew_assignments` and `mps_assignment`. We validate schema and IDs. To improve robustness we sample $k \in \{1, 3, 5\}$ diverse plans parallely and deduplicate them. While its possible for the LLM to provide infeasible actions, we haven't encountered this happen because of careful prompting requiring the LLM to choose from the set of feasible actions.

2.5 Simulation-Guided Selection

Each RC/MPS candidate proposed by the LLM is evaluated through a short rollout that follows the same physical model used in the precomputation phase, along with the greedy switching module described above. The simulation proceeds until the first reassignment trigger occurs, such as when an agent becomes idle or a planned action is completed. A short-horizon restoration score consistent with Eq (1) is then computed, and highest scoring candidate is selected, with ties resolved in favor of earlier proposals. The chosen candidate is executed step by step until an agent becomes idle, at which point a reassignment request is issued to the LLM.

2.6 Human-in-the-Loop Decision-Making

The planner allows real-time edits by human operators during post-disaster response. At any step, operators can modify crew speeds, fault durations, MPS limits, or block/clear roads reflecting updates common in emergency scenarios. We apply a state patch, recompute only affected affordances, and replan using LLM. Unlike MILP, this incurs no retraining or full re-solving overhead.

3 Experiments

We evaluate LARA on two standard distribution feeders: the IEEE 13-node test feeder, with 50 randomized scenarios, and the IEEE 33-node test feeder, with 10 representative scenarios. Each scenario defines fault locations and capacities, initial positions and speeds of RCs and MPSs, initial switch configurations, and load parameters (P, Q) with priority weights $W \in 1, \ldots, 10$.

For comparison, we implement the MILP following the approach of Lei (2019). This implementation serves as a reference baseline with a total inference time capped at one hour. We also adopt a multi-agent PPO baseline that employs a CTDE agents, incorporating action masking to filter infeasible operations. This baseline is referred to as *CTDE-RL*. Training (refer to A.1) is conducted separately on the IEEE 13-node and IEEE 33-node feeders using a single NVIDIA A100 GPU for 500k environment steps per feeder. The final checkpoints are selected based on validation rewards, and inference is performed on the CPU. Our planner is primarily executed on LLaMA-3.3-70B, with additional evaluations conducted on Gemini 2.5 Pro and Claude 3.7 Sonnet (Thinking). We perform ablation studies using $k \in \{1, 3, 5\}$ and

Feeder	Method	%MILP (mean)	Restoration Timesteps (mean)	Latency (median) s
IEEE-13	LARA (k=3)	95.58%	8.27	5.2
IEEE-13	RL (CTDE)	94.45%	6.51	0.5
IEEE-13	MILP (cap)	100.0%	4.44	5
IEEE-33	LARA (k=3)	112.99%	8	18.1
IEEE-33	RL (CTDE)	75.16%	22.5	0.5
IEEE-33	MILP (cap)	100.0%	14.6	58.8

Table 1: Performance comparison of methods across IEEE 13- and 33-node feeders.

Model (k=3)	IEEE-13 (%MILP)	IEEE-33 (%MILP)	Latency (s)
Llama 3.3 70B	95.58%	112.99%	25.2
Claude 3.7 Sonnet	95.99%	103.27%	120.6
Gemini 2.5 Pro	97.85%	96.80%	100.4

Table 2: Backbone sensitivity and latency at k=3 across LLMs (per-feeder mean score; higher is better).

temperature values $T \in \{0.2, 0.5, 0.8\}$ to assess the impact of proposal diversity. The primary evaluation metric is the cumulative weighted power restored. Specifically, let E_{LARA} and E_{MILP} denote the cumulative weighted power restored by LARA and MILP, respectively. The relative performance is then computed for each scenario as $(100 \times E_{\text{LARA}}/E_{\text{MILP}})$ before averaging across all scenarios. The secondary metrics include decision latency and the avg restoration time. Restoration timestep is defined as the total number of simulator steps required to fully energize all loads in the network.

3.1 Main Results

We compare methods on three metrics: %MILP (higher is better), restoration timesteps (lower is better), and median decision latency.

Table 1 summarizes LARA against a time-capped MILP reference and a CTDE-RL baseline on the IEEE 13- and 33-node feeders. By design, LARA satisfies all hard constraints; MILP is a high-quality reference within its cap, and CTDE-RL is a trained-policy baseline. On IEEE-33, LARA completes in 8 steps on average, fewer than time-capped MILP (14.6) and CTDE-RL (22.5), indicating better sequencing under scarce crews/MPSs and benefiting from greedy switching. On IEEE-13, LARA averages 8.27 steps vs. MILP's 4.44 and RL's 6.51; this is expected because our objective maximizes Eq 1 (earlier restoration), not pure makespan. In practice, LARA restores more high-priority load earlier, hence a higher reward despite a longer completion time and attains 95.6% of MILP on the reward function. Overall, the larger-feeder gains align with LARA's higher %MILP, while the small-feeder gap reflects the early-restoration vs. completion-step trade-off.

Setting (k, T)	IEEE-13 (% MILP)	IEEE-33 (% MILP)
(3, 0.2)	95.58	112.99
(3, 0.5)	94.52	110.32
(3, 0.8)	91.24	99.93
(1, 0.2)	95.26	112.67
(3, 0.2)	95.58	112.99
(5, 0.2)	96.07	103.83

Table 3: Ablation study. First 3 rows vary temperature T with fixed k=3; last three vary k with fixed T=0.2

To demonstrate portability of the interface, we run the base case with k=1 on three backbones (Llama 3.3 70B, Gemini 2.5 Pro, Claude 3.7 Sonnet (Thinking); the comparison is provided in Table 2. The models perform comparably and achieve near 95% on the IEEE 13-node test feeder, however, on the IEEE 33-node test feeder LLama3.3 70b clearly outperforms Claude and Gemini.

3.2 Ablations

We study two knobs of the method: (i) number of proposals $k \in \{1, 3, 5\}$ shows quality increases with diminishing returns; Increasing k from 1 to 3 provides a minor benefit in both IEEE 13-node test feeder and IEEE 33-node test feeder, however as we increase k further to 5 there does not seem to be an increase in IEEE 13-node test feeder and a minor increase in IEEE 33-node test feeder, this is because the LLM tends to give fairly similar results (since we are testing with T=0.2), the minor difference in IEEE 33-node test feeder performance can be attributed to selection of different set of MPS nodes in k=3 and k=5. (ii) testing temperature values 0.2,0.5,0.8 at k=3 shows marginal drop in the model performance between by T=0.2, and T=0.5 and a sharp drop when we increase T to 0.8. Our best model performance was achieved with k = 3 and temperature of 0.2.

4 Conclusion

We cast power distribution system restoration as language-conditioned planning over prevalidated affordances, separating hard feasibility from LLM reasoning. We compose joint RC MPS macro-actions that are safe by construction and scored under strict constraints. Results show that it approaches MILP quality on small cases (IEEE 13-node test feeder) and surpasses a time-capped MILP and CTDE-RL on larger ones while maintaining zero violations.

Limitations

Our evaluation is limited to IEEE 13-node test feeder and IEEE 33-node test feeder with a small number of crews and MPS; results may not generalize directly to much larger networks. The planner uses shallow look-ahead (single-step simulation) to satisfy strict time budgets, which can miss long-term dependencies that MILP captures. Additionally, all results are obtained in simulation; real-world deployment would require validating against operator practices, richer physical models, and communication delays.

References

2017. Proximal policy optimization algorithms. In *ICML Workshop / arXiv preprint arXiv:1707.06347*.

2018. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*.

2018. Value-decomposition networks for cooperative multi-agent learning. In *AAMAS (Extended Abstract) / arXiv:1706.05296*.

2019. Resilient disaster recovery logistics of distribution systems: Co-optimizing service restoration with repair crew and mobile power source dispatch. *IEEE Transactions on Power Systems*.

2021. The surprising effectiveness of MAPPO in cooperative multi-agent games. In *NeurIPS Deep RL Workshop / arXiv:2103.01955*.

2022. Improvement of distribution systems' resilience via operational resources and demand response. *Electric Power Systems Research*.

2024. Multiperiod distribution system restoration with routing: Repair crews, mobile electric vehicles, and networked microgrids. *IEEE Transactions on Power Systems*.

Michael Ahn, Anthony Brohan, Noah Brown, and 1 others. 2022. Do as i can, not as i say: Grounding language in robotic affordances. In *Robotics: Science and Systems (RSS)*.

M. E. Baran and F. F. Wu. 1989. Network reconfiguration in distribution systems for loss reduction and load balancing. *IEEE Transactions on Power Delivery*, 4(2):1401–1407.

Tao Ding, Zhaoyu Wang, Wei Jia, Bo Chen, Chen Chen, and Mohammad Shahidehpour. 2020. Multiperiod distribution system restoration with routing repair crews, mobile electric vehicles, and soft-open-point networked microgrids. *IEEE Transactions on Smart Grid*, 11(6):4795–4808.

Markus Freitag, David Grangier, Isaac Caswell, Colin Cherry, and 1 others. 2022. Minimum bayes risk decoding with neural metrics. *Transactions of the Association for Computational Linguistics*, 10:1086–1103.

Roshni Anna Jacob, Steve Paul, Souma Chowdhury, Yulia R. Gel, and Jie Zhang. 2024. Real-time outage management in active distribution networks using reinforcement learning over graphs. *IEEE Transactions on Smart Grid*.

William H. Kersting. 2001. Radial distribution test feeders. In *IEEE Power Engineering Society Winter Meeting*.

Jinsub Kim and Yury Dvorkin. 2019. Enhancing distribution system resilience with mobile energy storage and microgrids. *IEEE Transactions on Smart Grid*, 10(5):4996–5006.

Shunbo Lei, Chen Chen, Hui Zhou, and Yunhe Hou. 2019. Routing and scheduling of mobile power sources for distribution system resilience enhancement. *IEEE Transactions on Smart Grid*, 10(5).

Meysam Nazemi, Payman Dehghanian, Xiaonan Lu, and Chen Chen. 2021. Uncertainty-aware deployment of mobile energy storage systems for distribution grid resilience. *IEEE Transactions on Smart Grid*, 12(4):3200–3214.

Timo Schick, Jane Dwivedi-Yu, and 1 others. 2023. Toolformer: Language models can teach themselves to use tools. arXiv:2302.04761.

Feifan Shen, Qiuwei Wu, and Yusheng Xue. 2020. Review of service restoration for distribution networks. *Journal of Modern Power Systems and Clean Energy*, 8(1):1–14.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search for improved description of complex scenes. In *NeurIPS*.

Yi Wang, Dawei Qiu, Fei Teng, and Goran Strbac. 2024. Towards microgrid resilience enhancement via mobile power sources and repair crews: A multi-agent reinforcement learning approach. *Applied Energy*.

Xiaodong Yang, Chongbo Xu, Jinyu Wen, Youbing Zhang, Qiuwei Wu, Wenping Zuo, and Shijie Cheng. 2023. Cooperative repair scheduling and service restoration for distribution systems with soft open points. *IEEE Transactions on Smart Grid*, 14(3):1827–1841.

Shunyu Yao, Jeffrey Zhao, and 1 others. 2023. React: Synergizing reasoning and acting in language models. arXiv:2210.03629.

Shuyu Yao, Peng Wang, Xiaoyu Liu, Hong Zhang, and Tianshu Zhao. 2020. Rolling optimization of mobile energy storage fleets for resilient service restoration. *IEEE Transactions on Smart Grid*, 11(2):1030–1043.

Zhigang Ye, Chen Chen, Bo Chen, and Kai Wu. 2021. Resilient service restoration for unbalanced distribution systems with distributed energy resources by leveraging mobile generators. *IEEE Transactions on Industrial Informatics*, 17(2):1386–1396.

A Appendix

A.1 CTDE-RL Baseline Details

We implement a centralized-training, decentralized-execution (CTDE) PPO baseline ("CTDE-RL"). A centralized critic consumes a compact global state embedding, while decentralized actors—one each for RC, MPS, and switches produce actions for their respective agents. Illegal actions are masked before sampling to ensure feasibility.

For comparability with LARA, the same simulator and physics are used. The environment enforces radiality at every step and hard-checks MPS real/apparent-power limits. Episodes terminate when all faults are repaired or when a fixed step cap is reached, identical to the setting for the other methods.

Action spaces are discrete. The RC actor selects a crew-to-fault assignment or an idle/continue choice; faults that are unreachable or already completed are masked. The MPS actor selects a connection node together with an index to a feasible load subset; options violating $P_{\max}$ or $S_{\max}$ are masked. Unlike LARA, switching is learned: a dedicated switch actor chooses a legal switch toggle (or no-op) among currently open, cycle-free candidates, with illegal closures masked by the environment.

Observations are factorized by agent and normalized online. Each RC observes its position, speed, repair rate, current assignment, distances to a shortlist of feasible faults, and the remaining work for that shortlist. Each MPS observes its position, travel speed, $(P_{\max}, S_{\max})$, remaining energy, and a shortlist of feasible connection options (node identifier plus a summary of the load subset). The critic additionally receives global summaries of all damage points (locations and remaining work), a summary of energized load, and a compact reachability fingerprint induced by the current switch and fault configuration.

The per-step reward equals the newly energized weighted load $\sum_{n \in \Delta L_t} P_n W_n$, i.e., the exact increment of the reward objective; no additional shaping is used. Networks are multilayer perceptrons with hidden widths 256, 128, and 64 and ReLU activations; hidden layers use layer normalization, and categorical identifiers (e.g., fault or connection indices) are embedded and concatenated to continuous features.

Optimization uses PPO with the clipped surrogate objective and Adam. Unless otherwise noted, $\gamma = 0.99$, $\lambda = 0.95$, clip ratio 0.2, learning rate 3×10^{-4}, entropy coefficient 0.01, value-loss coefficient 0.5, gradient-norm clip 0.5, rollout length 128, four minibatches, and four epochs per update, with N parallel environments chosen to keep the effective batch size stable. We train separately on the IEEE-13 and IEEE-33 feeders for 500k environment steps each on a single NVIDIA A100 GPU, select checkpoints by validation reward, and evaluate on CPU with masked greedy (argmax) action selection.

Separate training per feeder is required because the observation and action cardinalities depend on feeder topology and scale (numbers of nodes, switches, feasible connection options, and masking sets). A single shared policy would face mismatched heads and distribution shift across feeders, which empirically degrades transfer.

Scenario splits are identical across methods: CTDE-RL uses training and validation scenarios to fit and select checkpoints, and all methods are finally evaluated on the same held-out test scenarios reported in Section 3.1. The underlying constraints and physics are identical; only the switching decision differs, being learned in CTDE-RL and greedy in LARA.

B Sample Topologies

B.1 IEEE 33-Node Test Feeder

We use the standard radial 33-node feeder with a single source and priority-weighted loads ($W \in \{1, \ldots, 10\}$). Scenarios contain 6–8 simultaneous faults, 2 repair crews (RCs), and 2 mobile power sources (MPSs). RCs move at finite speed and repair at a fixed per-step rate; each MPS travels, then spends one connection step and may supply any feasible subset of local loads subject to its $P_{\max}$, $S_{\max}$, and finite energy budget. Switch actions are set by the greedy reconfiguration module and always preserve radiality. We evaluate 10 representative scenarios with randomized fault placements/workloads, initial RC/MPS positions and speeds, and load priorities. Episodes terminate when all faults are repaired and no island remains without grid or MPS supply. See Figure 2 for the feeder diagram.

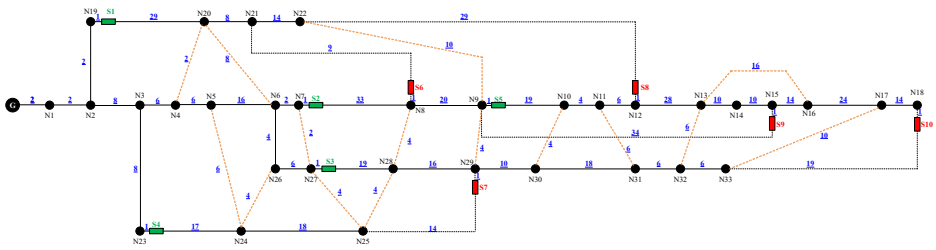

Figure 2: IEEE 33-node test feeder diagram (S1–S10 are switches).

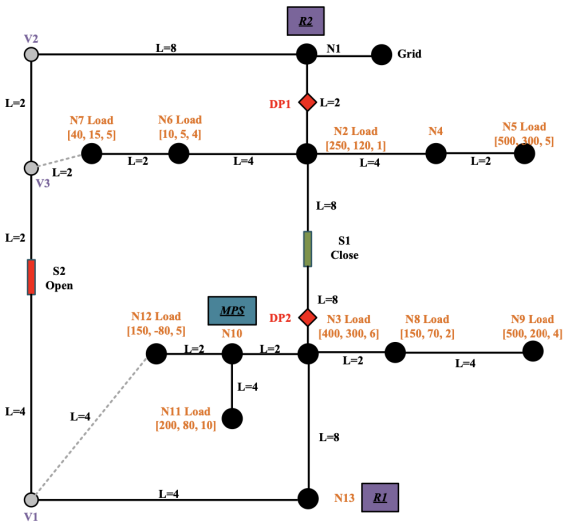

Figure 3: IEEE 13-node test feeder diagram (R1, R2 are RCs; S1, S2 are switches).

B.2 IEEE 13-Node Test Feeder

We also evaluate on the radial 13-node feeder with the same priority scheme. Scenarios use 2–5 simultaneous faults, 2 repair crews, and 1 MPS. Agent kinematics and MPS feasibility are identical to the 33-node setup; switches are again chosen by the greedy, radiality-preserving module. We generate 50 randomized scenarios by sampling fault locations/workloads, initial agent positions/speeds, and load priorities. Episodes end once all faults are repaired and no island remains unserved. The network diagram is shown in Figure 3.