Resilience-Aware Controlled Islanding via a Multi-Objective QAOA Pipeline
(Rigetti Hardware Validation & Large-Scale Classical Scaling)

Team: QuantumPulse
Rushikesh Ubale ; Yasar Mulani;Vashti Chowla; and Aya Rhazi

February 28, 2026

Abstract

Modern power transmission networks face unprecedented operational stress from high renewable penetration, variable loads, and concentrated demand (e.g., AI datacenters). Operators need tools that quickly identify robust network partitions (controlled islanding) which limit cascading outages while preserving maximum transferable power. We propose a resilience-aware multi-objective formulation of the Maximum Power Energy Section (MPES) problem and a hybrid quantum-classical pipeline based on QAOA for hardware-validated small cores and classical decomposition for full-scale networks. We demonstrate experimental plans for Rigetti hardware (Problem A: 21 nodes, 28 edges) and large-scale classical benchmarks on a 180-node public instance (Problem B). The document details motivation, formal problem statement, algorithmic strategy, evaluation metrics, and a reproducible experimental roadmap.

Contents

1 Introduction and Motivation

Electrical grids are becoming more volatile due to high shares of renewables, extreme weather, and new concentrated loads. In emergencies, grid operators may intentionally partition the system ("controlled islanding") to prevent larger cascading blackouts. A natural computational proxy for the partitioning problem is a weighted Maximum Cut: identify a bipartition of nodes that maximizes aggregate edge admittance across the cut — the Maximum Power Energy Section (MPES). However, the practical operational desiderata require balancing transfer maximization with resilience (avoid cutting critical backbone edges, avoid fragmentation into tiny islands, and maintain operational feasibility). We develop a multi-objective QUBO/Ising formulation targeting these goals, and a hybrid pipeline that uses quantum hardware where it is most impactful (dense or critical cores) and classical solvers for large-scale stitching.

2 Datasets and Problem Instances

We use two graph instances (edge lists) derived from a public South Carolina transmission subset. Each row is an edge (i, j, w_{ij}) where w_{ij} denotes a physics-derived link weight (admittance proxy).

Problem	Nodes	Edges
Problem A (small, hardware candidate)	21	28
Problem B (large, scalability)	180	226

Table 1: Dataset summary (edge lists).

Basic EDA findings (computed): the full graph is connected; Problem B is sparse (density ≈ 0.014) and modular (community structure). Problem A admits exact solution via brute force (exact MPES reported: 3728.4132) and supports subgraph extraction for hardware validation.

3 Research Gaps & Related Work

3.1 Typical limitations of existing quantum grid work

A review of the quantum optimization literature for power systems shows common limitations:

- evaluation primarily on toy IEEE networks or small random graphs (not real transmission topologies);
- single-objective MaxCut formulations that can produce operationally infeasible partitions (tiny islands, cut of critical backbone links);
- limited demonstrations on real quantum hardware and weak scaling studies to large, realistic networks.

Representative references include: Dupont et al. ("Optimization via Quantum Preconditioning") which argues for hybrid preconditioning strategies; surveys of QAOA scaling; and power-systems research on controlled islanding. These motivate a problem formulation that is both operationally meaningful and QUBO compatible.

4 Mathematical Preliminaries and Notation

4.1 Graph-Theoretic Notation

Let $G = (V, E, w)$ denote a weighted, undirected graph where:

- $V = \{1, 2, \ldots, n\}$ is the set of nodes (buses/substations), $|V| = n$.
- $E \subseteq V \times V$ is the set of undirected edges (transmission lines).
- $w : E \to \mathbb{R}_{>0}$ assigns a positive weight w_{ij} to each edge $(i, j) \in E$, representing line admittance (electrical strength).

We define the weighted adjacency matrix $W \in \mathbb{R}^{n \times n}$ such that:

$$W_{ij} = \begin{cases} w_{ij}, & (i,j) \in E,\ 0, \\ \text{otherwise.} \end{cases} \tag{1}$$

For each node $i \in V$, denote its degree by $\deg(i)$ and its weighted degree by $\deg_w(i) = \sum_{j:(i,j)\in E} w_{ij}$.

4.2 Partition Variables

We represent a bipartition of G using spin variables

$$z = (z_1, \ldots, z_n) \in \{\pm 1\}^n, \tag{2}$$

where $z_i = +1$ if node i belongs to partition A and $z_i = -1$ otherwise. The vector z uniquely defines a cut $(A, V \setminus A)$.

5 Problem Formulation

5.1 Graph and baseline objective

Let $G = (V, E)$ be a weighted undirected graph, $|V| = n$, with weights $w_{ij} > 0$ for $(i, j) \in E$ (edge admittance proxy). For spin variables $z_i \in \{\pm 1\}$ the weighted MaxCut objective (MPES) is:

$$C(z) = \frac{1}{2} \sum_{(i,j)\in E} w_{ij}(1 - z_i z_j). \tag{3}$$

Maximizing $C(z)$ selects a partition that maximizes the total admittance across the cut.

5.2 Multi-objective augmentation

Plain MaxCut is insufficient for operational resilience. We augment $C(z)$ with penalties designed to enforce balance, limit fragmentation, and protect critical backbone edges while preserving quadratic structure (2-local) so the problem remains QUBO/Ising compatible.

5.2.1 Criticality-weighted edges

Compute an edge criticality score C_{ij} (e.g., edge betweenness centrality or flow centrality). Form the adjusted weight:

$$\tilde{w}_{ij} = w_{ij} \cdot \bigl(1 + \beta\, C_{ij}\bigr), \tag{4}$$

where $\beta \geq 0$ controls emphasis on backbone preservation. The new core term is $C_{\tilde{w}}(z)$, same form as (3) with $\tilde{w}$.

5.2.2 Balance penalty

To discourage extremely imbalanced partitions we add a quadratic penalty:

$$P_{\text{bal}}(z) = \lambda_{\text{bal}} \left(\sum_{i\in V} z_i\right)^2. \tag{5}$$

This is 2-local when expanded and thus implementable in QUBO.

5.2.3 Fragmentation surrogate

A direct count of small connected components is not quadratic. We propose a quadratic surrogate that penalizes local disagreement weighted by nodal vulnerability:

$$P_{\text{frag}}(z) = \lambda_{\text{frag}} \sum_{(i,j)\in E} \alpha_{ij}(1 - z_i z_j), \tag{6}$$

where α_{ij} is chosen to increase penalty when cutting edges that would isolate low-degree nodes (e.g., $\alpha_{ij} = 1/(\deg(i) + \deg(j))$ or another normalized vulnerability factor). This remains 2-local and blends with the core cut metric.

5.2.4 Total QUBO objective

We introduce $F(z)$ to be maximized (or equivalently minimize $-F$ in QUBO form):

$$F(z) = C_{\tilde{w}}(z) - P_{\text{bal}}(z) - P_{\text{frag}}(z). \tag{7}$$

All terms are quadratic (or approximated as quadratic) so conversion to an Ising Hamiltonian is straightforward. Proper scaling and tuning of λ parameters is critical and will be done via sensitivity sweeps on classical solvers prior to hardware submission.

6 Algorithmic Pipeline

We propose a hybrid divide-and-conquer approach that is practical on current NISQ hardware:

1. **Preprocessing and scoring:** compute edge criticality C_{ij}, degree statistics, and normalize w_{ij}. Produce $\tilde{w}_{ij}$ and vulnerability weights α_{ij}.
2. **Decomposition:** split the large graph (Problem B) using edge-betweenness divisive clustering or spectral community detection into connected subgraphs of target size k (recommend $k \in [8, 12]$ for Rigetti hardware experiments).
3. **Local QUBO solve:** for each subgraph, solve the multi-objective QUBO with a hybrid approach: (a) classical simulator optimization to tune angles for shallow QAOA (p=1 or p=2), (b) optionally run these optimized angles on Rigetti for select subgraphs (Problem A or cores) using precompiled circuits and sampling only.

4. **Stitching / boundary optimization:** resolve boundary edges between subgraphs with a small classical local search (`one_exchange()` greedy heuristic in NetworkX) or with a compact boundary QUBO restricted to interface nodes only.

5. **Evaluation and iteration:** compute metrics (approximation ratio, island sizes, resilience scores) and iterate penalty weights or decomposition strategy as needed.

7 Resource Estimation (Big-O Practical Limits)

For a standard one-node-per-qubit QAOA with p layers:

- Qubit count: n (nodes). Problem B would require $n = 180$ qubits.
- Two-qubit gates per cost layer: $\mathcal{O}(|E|)$; total two-qubit gates $\mathcal{O}(p|E|)$.
- Single-qubit gates: $\mathcal{O}(pn)$.
- Embedding overhead: nonlocal logical edges require SWAPs on nearest-neighbor superconducting topologies, multiplying two-qubit count by factor S; compiled depth grows as $\Omega(p \cdot S \cdot \chi(G))$.

These reasons make direct QAOA infeasible on Problem B for NISQ devices; our pipeline avoids this by running quantum circuits only on small subgraphs.

8 Benchmarking and Evaluation Metrics

Baselines: greedy one-exchange (NetworkX), simulated annealing, spectral partition, and Goemans–Williamson SDP relaxation where feasible. We report:

- Approximation ratio $r = F_{\text{alg}}/F_{\text{opt}}$ (exact for Problem A; relative to best classical for Problem B).
- Island quality (component sizes, count).
- Resilience measures (post-cut largest component, N-1 vulnerability statistics).
- Runtime and wall-clock costs.

9 Hardware Execution Plan (Rigetti)

Target small connected subgraphs (8–10 nodes) for Rigetti Ankaa devices. Use pyQuil and QCS SDK. Steps:

1. Produce Ising coefficients (ZZ coefficients, local Z terms if any) and compile into native RX/RZ/iSWAP gate sequences.
2. On classical simulator, tune (γ, β) via grid search or COBYLA at p=1 (p=2 optional).
3. Submit hardware job(s) with precomputed angles and sample ≥ 1000 shots per job. Use readout mitigation and postselection where possible.
4. Aggregate samples, compute empirical cut values, and compare to simulator and classical baselines.

10 Risk Analysis and Mitigation

- **Noise and fidelity limits:** Keep circuits shallow and two-qubit counts under ~ 100 when possible; precompile and use mitigation.
- **Penalty tuning sensitivity:** perform classical sweeps on small subgraphs to identify stable parameter windows.
- **Decomposition error:** compare multiple decomposition heuristics and use boundary QUBOs to improve stitching.

11 Experimental Roadmap and Deliverables

We propose a 48-hour hackathon schedule with deliverables:

1. Reproducible code + Jupyter notebooks for EDA and formulation.
2. QUBO builder parameter sweep notebooks.
3. QAOA simulator results (p=1, p=2 optional) and plots (approx ratio vs p, vs subgraph size).
4. Rigetti pyQuil submission notebooks, hardware logs, and analysis (shots, distributions).
5. Large-scale classical benchmark on Problem B and a comparative report.
6. Final 2-page executive summary and presentation slides.

12 Conclusion

This project bridges a clear research gap between quantum optimization algorithms and industry needs for transmission resilience. The multi-objective QUBO keeps the problem in the quadratic domain while encoding resilience and operational constraints, and the hybrid divide-and-conquer pipeline is engineered to leverage NISQ devices where they matter most while producing scalable, actionable results for real networks.

References

References

[1] M. Dupont, T. Oberoi, B. Sundar, "Optimization via Quantum Preconditioning," arXiv:2502.18570 (2025).

[2] Review article on QAOA scaling and limits (survey). Example: A. Author, "QAOA: promises and limitations", Journal of Quantum Computing (2024).

[3] NetworkX documentation: edge$_{b}etweenness_{p}artition$,

Rigetti QCS documentation and Ankaa device notes, `https://qcs.rigetti.com/qpus`

Controlled islanding literature: e.g., effective islanding and operational constraints discussion. (See power systems community literature.)

A Appendix: Practical QUBO mapping notes

For completeness: writing the Ising Hamiltonian from (7) implies mapping $z_i \in \{\pm 1\}$ to binary $x_i \in \{0, 1\}$ if needed, or directly using Pauli-Z operators on qubits. The cost layer implements ZZ rotations with angles proportional to coefficients; mixer uses RX rotations. Keep p small and control two-qubit gate count.