Practical Evaluation of Quantum-Based Grid Partitioning for Parallel Simulation

Carsten Hartmann
Energy System Engineering (ICE-1)
Forschungszentrum Jülich
52428 Jülich, Germany
0009-0007-5067-589X

Arnau Chaparro Arimon
Technische Universität Berlin
10623 Berlin, Germany
0009-0009-2716-9109

Junjie Zhang
Energy System Engineering (ICE-1)
Forschungszentrum Jülich
52428 Jülich, Germany
0000-0003-1720-028X

Thiemo Pesch
Energy System Engineering (ICE-1)
Forschungszentrum Jülich
52428 Jülich, Germany
0000-0002-3297-6599

Carlos D. Gonzalez Calaza
Jülich Supercomputing Centre (JSC)
Forschungszentrum Jülich
52428 Jülich, Germany
0000-0001-8467-8633

Andrea Benigni
Energy System Engineering (ICE-1)
Forschungszentrum Jülich
52428 Jülich, Germany
0000-0002-2475-7003

***Abstract*—Optimal grid partitioning is essential for efficient parallel simulation. In a previous work, we formalized the optimal partitioning problem as a multi-objective optimization task and introduced a corresponding QUBO formulation suitable for quantum computing. In this work, we evaluate this approach in a practical simulation workflow setting using an established parallel solver for dynamic power grid simulations. Our results demonstrate that the QUBO objective for grid partitioning correlates with the actual simulation time in a parallel simulator, validating the suitability of the QUBO formulation. Furthermore, we analyze the trade-off between longer optimization runs (i.e., more samples) and overall simulation performance.**

***Index Terms*—Parallel algorithms, Power system simulation, Quantum Computing, Graph theory**

I. Introduction

The transition to renewable energy has greatly increased the complexity of modeling, simulating, and controlling modern power systems [1]. This calls for high-fidelity yet computationally efficient simulation methods capable of capturing fast system dynamics and evaluating a wide range of operating conditions [2].

However, detailed time-domain simulations are computationally intensive, and scaling them to large networks remains a key challenge [3]. For operational tasks such as contingency analysis, where many scenarios must be assessed rapidly, even modest computational speed-ups can provide significant practical benefits [4], [5].

One promising route toward greater scalability is parallelisation: the power network is divided into smaller subsystems that can be simulated concurrently on separate computing nodes. Considering the specific algorithm [6], an optimal partitioning should balance computational loads and minimize communication overhead between subsystems. In [7], we formalize the optimal partitioning problem as a multi-objective optimization task and introduce a corresponding quadratic unconstrained binary optimization (QUBO) formulation compatible with current quantum hardware, such as the D-Wave Advantage System [8]. The work illustrates how quantum resources can be integrated into classical HPC workflows to enhance specific computational steps, thereby circumventing existing algorithmic and hardware constraints that currently prevent full dynamic power grid simulations from being executed on quantum devices. In this context, we focus on the theoretical foundations of the proposed QUBO formulation and provide a comprehensive QPU hardware analysis, including hardware requirements and noise characteristics.

In this article, we provide a practical evaluation of the quantum-based partitioning in the simulation, using a publicly available parallel diacoptics-based solver [9]. Since the solver architecture does not allow manual component allocation to computational nodes, the tested QUBO does not capture the actual implementation completely. Nevertheless, under these limitations, we establish that the QUBO objective values correlate with actual simulation times for two IEEE test cases, validating our previous theoretical considerations. Furthermore, we analyze the trade-off between longer optimization, practically generating larger sample sets, and total simulation time of the workflow.

II. Review: QUBO Formulation for Grid Partitioning for Parallel Simulation

We briefly summarize the key concepts and formulation steps from [7]. Following the algorithms in [6], [10], [11], a parallel simulation step of a partitioned grid can be divided

The paper was written as part of the project "Quantum-based Energy Grids (QuGrids)", which is receiving funding from the programme "Profilbildung 2022", an initiative of the Ministry of Culture and Science of the State of North Rhine-Westphalia. The authors gratefully acknowledge Prof. Kristel Michielsen and the Jülich Supercomputing Centre (https://www.fz-juelich.de/en/ias/jsc/systems/supercomputers/apply-for-computing-time/juniq) for funding this project by providing computing time on the D-Wave Advantage™ System JUPSI through the JUNIQ project, funded by the German Federal Ministry of Research, Technology and Space (BMFTR) and the Ministry of Culture and Science of the State of North Rhine-Westphalia.

979-8-3315-8138-1

TABLE I: Optimization objectives for optimal partitioning.

Stage	Simulation Task	Objective
1	Solve component equations within each sub-network to obtain injections.	Minimize idle time due to unequal component allocations.
2	Solve component and network equations across the cut.	Minimize cut size.
3	Solve network equations in parallel using component and cut injections.	Minimize idle time from unbalanced sub-network sizes.

into three stages, each defining an optimization objective for the partitioning task, c.f. Table I.

Electrical networks can be represented as graphs $\mathcal{G}_{\text{grid}} = (\mathcal{V}, \mathcal{E})$, where nodes are buses and edges are electrical connections. The partitioning objectives (Table I) are thus mapped to a graph partitioning problem, which is NP-hard [12]. Encoding node allocations as binary variables z_n ($z_n = 0$ if $n \in \mathcal{V}_1$, $z_n = 1$ if $n \in \mathcal{V}_2$) yields a natural QUBO formulation, c.f. also [13].

The mapping proceeds in two steps. We first start from the graph $\mathcal{G}_{\text{grid}}$ and build a graph $\mathcal{G}_{\text{comp}}$ that unifies the electrical component representation. In particular, we also represent all components that are not electrical connections, e.g. generators, by additional edges that are incident to the node where the component is placed. Furthermore, we assign edge weights c_e to all edges $e \in \mathcal{E}_{\text{comp}}$ that reflect the computation cost, e.g. in floating point operations (FLOPs), of solving the respective component equations. Second, we derive QUBO objectives for each simulation stage, leading to the total objective

$$Q_{\text{part}} = Q_{\text{comp}} + Q_{\text{cut}} + Q_{\text{net}}. \tag{1}$$

Using the graph representations $\mathcal{G}_{\text{grid}}$ and $\mathcal{G}_{\text{comp}}$, we obtain

$$Q_{\text{comp}} = \left(\sum_{n \in \mathcal{V}_{\text{grid}}} \alpha(n) z_n - \frac{W}{2} \right)^2, \tag{2}$$

$$\begin{aligned} Q_{\text{cut}} =& \frac{1}{2} \sum_{n,m \in \mathcal{V}_{\text{grid}}} A_{n,m}(\mathcal{G}_{\text{grid}}) \left(c_{n,m} + 4N - 1\right) \\ & (-2 z_n z_m + z_n + z_m), \end{aligned} \tag{3}$$

$$Q_{\text{net}} = (2N-1)^2 \left(\sum_{n \in \mathcal{V}_{\text{grid}}} z_n - \frac{N}{2} \right)^2, \tag{4}$$

where $\alpha(n) = \frac{1}{2} \sum_{m \in \mathcal{V}_{\text{grid}}} c_{n,m} + \sum_{m \in \mathcal{V}_{\text{comp}} \setminus \mathcal{V}_{\text{grid}}} c_{n,m}$, $W = \sum_{n \in \mathcal{V}_{\text{grid}}} \alpha(n)$, and $N = |\mathcal{G}_{\text{grid}}|$ is the number of buses. Each objective is designed to give a realistic estimation of the computational overhead, e.g. in the network step the difference in the computational resources required for the forward and backward substitution when using LU decomposition.

The optimal partitioning for the IEEE 39-bus test case is shown in Fig. 1. Grey nodes indicate the added auxiliary nodes representing generators. The resulting partition achieves balanced sub-network sizes and generator distributions while minimizing the cut set, aligning with the intended objectives.

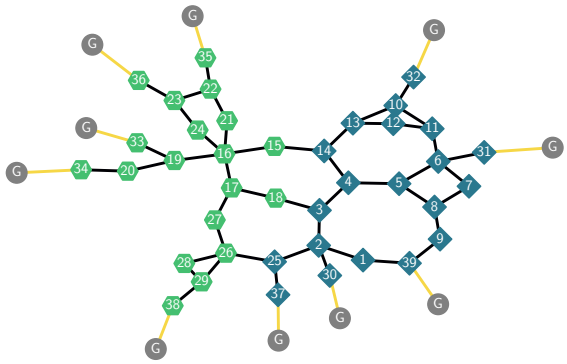

Fig. 1: Optimal Partioning for Case 39.

III. Experimental Setup

A. Sampling Partitions on D-Wave QPU

Quantum annealing seeks the ground state of a problem Hamiltonian by adiabatically transforming a quantum system from an initial, easily prepared Hamiltonian to the target problem Hamiltonian, ideally remaining in the ground state throughout the evolution [14], [15]. The D-Wave Advantage system implements this using a programmable Ising Hamiltonian, allowing QUBO problems, which are mathematically equivalent to Ising models, to be solved on its quantum processing unit (QPU).

In practice, the evolution is not perfectly adiabatic, and the system may transition to excited states, making D-Wave a heuristic rather than deterministic solver. Two parameters primarily determine performance: the *annealing time* T_{A} and the *number of anneals* N_{S}. According to the adiabatic theorem, longer T_{A} increases ground-state probability [16], but also increases exposure to decoherence and thermal noise, requiring empirical tuning. The total QPU access time is approximated as

$$T_{\text{QPU}} = N_{\text{S}} T_{\text{A}}, \tag{5}$$

neglecting overheads [17].

For each sampled configuration with objective value E_{sampled}, we define the relative error

$$L = \frac{|E_{\text{min,global}} - E_{\text{sampled}}|}{|E_{\text{min,global}}|}, \tag{6}$$

where $E_{\text{min,global}}$ is the optimal value obtained with a deterministic solver such as Gurobi. The *lowest relative error* [8],

$$L_{\text{min}} := \min_{E_{\text{sampled}}} L, \tag{7}$$

quantifies the best solution quality within a sample set.

Beyond T_{A} and N_{S}, solution quality depends strongly on the QUBO embedding and chain strength [7], [18]. We therefore perform a grid search over ten embeddings and eight log-uniformly spaced chain strengths in $[0.02, 0.75]$ with $T_{\text{A}} = 20\ \mu\text{s}$ and $N_{\text{S}} = 10^5$. The best configuration is then

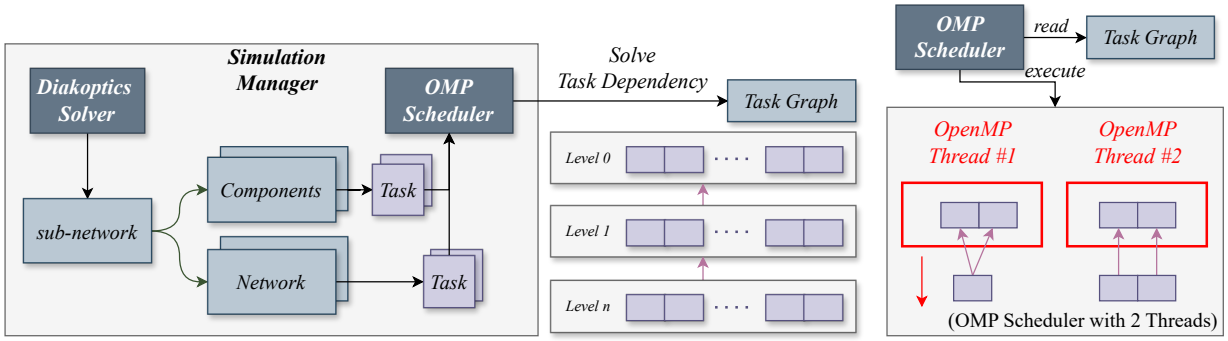

Fig. 2: Automatic Task Scheduling in DPsim.

sampled $N_{\mathrm{S}} = 10^6$ times for $T_{\mathrm{A}} \in \{1, 50, 200\}$ μs to generate large sample sets for the total workflow evaluation.

B. Parallel Simulation of Partitioned Grids in DPSim

The publicly available simulator DPsim [9] is employed in the experiment to verify partition qualtity and the overall workflow. Its internal computing task scheduling is shown in Fig. 2. Computations such as numerical integration for each component, solution of network equations, are encapsulated as `task`. When the Diakoptics solver generates these tasks, it also encodes their dependencies. Before the simulation begins, the simulation manager forwards all tasks to the OpenMP scheduler, which automatically constructs a task graph from the dependencies. Tasks on the same level of this graph can then be executed in parallel. In DPsim, the scheduler assigns tasks to N_{omp} OpenMP threads via static dispatch for parallel execution. If the allocated threads are larger than the number of partitions, sub-networks can be processed according to the partitioning.

However, note that in this practical setup, the component tasks are assigned automatically to theOpenMP threads, and thus independently of the network partitioning obtained from the QUBO minimization. That is, the QUBO (1), particularly the influence of the component balancing, can not be fully tested in this setup. Moreover, in the current DPsim release, transformers cannot be recognized in the cut. As a result, not all possible cuts can be tested; only branches, i.e., transmission lines, are considered.

The simulations were carried out on a desktop workstation equipped with an AMD Ryzen 7 3700X 8-core CPU @ 3.6 GHz and 16 GB RAM. DPsim was executed within Windows Subsystem for Linux (WSL 2) on Windows 10 Pro (64-bit) using Ubuntu 24.04.1. Given a valid partitioning, we average the simulation time T_{sim} over 50 simulation runs to decrease the influence of the OpenMP thread allocation, as well as other computational loads on the CPU. We also run 10 warm-up runs, which we discard, as this increases robustness.

Lastly, the comparison between parallel simulation using Diakoptics solver and sequential simulation is shown in Fig. 3. The relative error between two voltage signals, given by $1 - \frac{v_{n,\mathrm{par}}(t)}{v_{n,\mathrm{seq}}(t)}$ where $v_{n,\mathrm{par}}(t)$ is the voltage at node n in parallel simulation and $v_{n,\mathrm{seq}}(t)$ in sequential, are negligible.

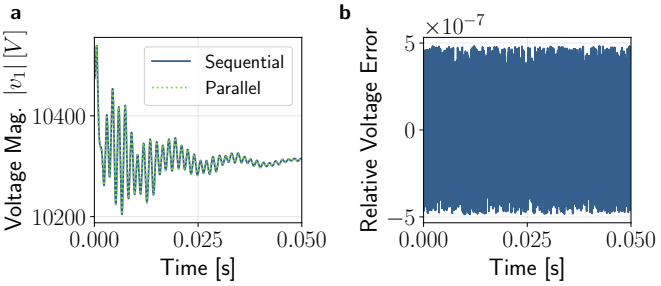

Fig. 3: Verification of parallel simulation for Case 39. A small disturbance is set at the beginning of the simulation. **a** Comparison of nodal voltages at node 1. **b** Relative voltage error is negible at node 1.

C. Workflow Evaluation Setup

The total workflow consists first of the quantum-based partitioning of the grid and second, dynamic simulation on classical hardware. Let the total simulation time be defined as

$$T_{\mathrm{total}} = T_{\mathrm{QPU}}(N_{\mathrm{S}}, T_{\mathrm{A}}) + T_{\mathrm{sim}}. \tag{8}$$

Hence, a trade-off arises between investing additional effort in quantum sampling, by increasing N_{S} or T_{A}, and accepting a suboptimal partition that is cheaper to obtain but more costly to simulate.

To efficiently explore this trade-off, smaller sample subsets are randomly drawn from the large precomputed sample set ($N_{\mathrm{S}} = 10^6$) to emulate multiple, smaller QPU runs. This approach conserves valuable quantum hardware access time while maintaining statistically representative results for the trade-off analysis.

IV. Results

A. Simulation Times vs QUBO Objective

A comparison between the full QUBO objective values (1) and the actual parallel simulation times T_{sim} obtained from DPsim reveals a clear relationship between energy and runtime performance. In particular, lower QUBO energies (i.e., lower relative errors L) generally correspond to shorter average simulation times for Case 39 using 4 OpenMP threads, and for Case 118 using both 2 and 4 OpenMP threads, cf. Fig. 4.

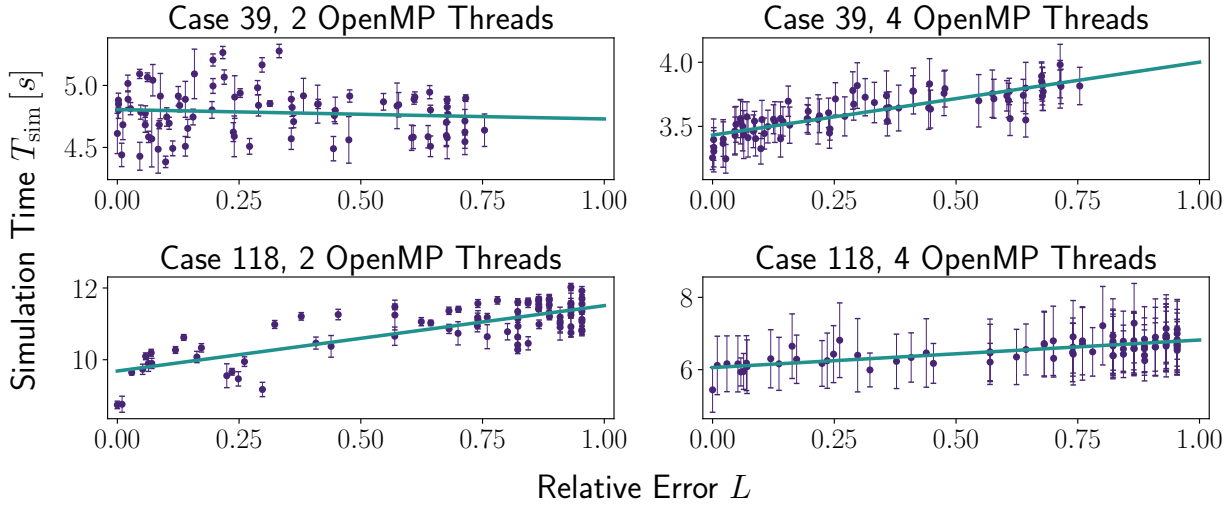

Fig. 4: Simulation Time T_{sim} vs Relative Error L for Q_{part}, c.f Eq. (1). Error bars represent the standard deviation across 50 simulation runs, while green lines indicate linear trend fits.

For Case 39 with 2 OpenMP threads, however, no significant correlation between QUBO energy and simulation time is observed. For such smaller test cases, the computational workload from component models is distributed relatively evenly across the available threads, such that for a small number of threads, the component-solving tasks dominate the total workload. This leads to relatively uniform performance across different L. However, as the number of threads increases, the component workload per thread decreases, and the network solve becomes the dominant part of the computation. Consequently, performance becomes more sensitive to the optimization error, as imbalanced partitions have a greater impact when the network solve dominates.

In contrast, for Case 118, the speed-up achieved with 2 OpenMP threads is more pronounced than with 4 threads. Interestingly, when using 4 threads, the variance in simulation time T_{sim} increases significantly. This behavior can be attributed to several factors: (i) increased cache misses due to higher memory contention, (ii) more frequent context switching and synchronization overhead, and (iii) imbalanced workload distribution among threads in the automatic scheduling of parallel regions.

Finally, we note two things: First, for smaller benchmark cases, no substantial speed-ups are observed, likely because the computational workload is too small to benefit meaningfully from thread-level parallelization. Second, for a simplified QUBO without component balancing, and thus more aligned with DPSim, a similar correlation between QUBO objective values and simulation times is observed (not shown here).

B. Workflow Evaluation

A trade-off exists between the effort invested in the optimization phase, which is controlled by the sample size N_{S} and the annealing time T_{A}, and the resulting total simulation time T_{total}, cf. Fig. 5. Here, we investigate this behavior for Case 39, where all dynamic power system simulations are performed using four OpenMP threads. We observe that T_{total} as a function of N_{S} exhibits a skewed, double-minimum ("W-shaped") behavior.

For small N_{S}, the probability of obtaining low-energy (and hence high-quality) partitionings is low, leading to suboptimal performance in the subsequent simulation. As N_{S} increases, this probability improves significantly, and correspondingly, better partitionings reduce the overall simulation time. However, beyond a certain point, the computational overhead of the quantum optimization, which scales approximately linearly with N_{S}, begins to dominate. Consequently, for large N_{S}, the total time T_{total} increases again, and its variance decreases, as good solutions are sampled more consistently.

For short annealing times ($T_{\mathrm{A}} = 1\,\mu\mathrm{s}$), we find that the optimal sample size is around $N_{\mathrm{S}} \gtrsim 10^3$. In contrast, for longer annealing times, the optimal N_{S} decreases. This trend is driven by two combined effects: (i) according to the adiabatic theorem [16], longer T_{A} increases the likelihood of remaining in or reaching the ground state, and (ii) the QPU access time scales linearly with both T_{A} and N_{S}, such that its contribution to T_{total} becomes particularly dominant for large T_{A}.

The results indicate that, under current hardware conditions, the most efficient strategy is to use very short annealing times and moderately large sample sets ($N_{\mathrm{S}} \sim 10^3$), effectively performing a form of fast, biased random sampling [19], [20]. However, with future hardware improvements, in particular, faster annealing controls, reduced noise, and lower QPU overheads, the regime of longer annealing times combined with smaller N_{S} is expected to become favorable.

V. Discussion

Our results show that optimizing partitioning using a quantum annealer can, on average, accelerate parallel simulations.

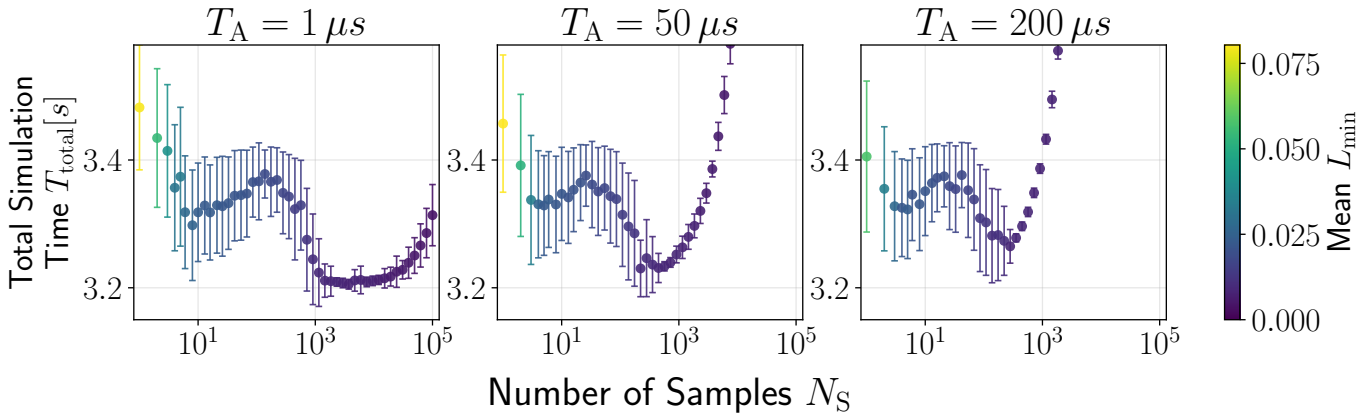

Fig. 5: Trade-off between total simulation time T_{total} and sample set size N_{S}, as well as annealing time T_{A} for Case 39 simulated with 4 OpenMP Threads. Error bars represent the standard deviation across 100 randomly drawn sample subsets.

Furthermore, we identified a clear trade-off between quantum sampling effort and efficiency: small sample sets or short annealing times are fast but often produce suboptimal partitions, while larger sets or longer anneals improve partition quality but eventually incur overhead that diminishes returns. For the small case studied, the optimal strategy combines short annealing times with moderately large sample sets, balancing QPU effort and simulation savings. For larger test cases, this trade-off needs to be revisited as hardware improves, c.f. our discussion in [7]. Moreover, our analysis neglected overheads such as classical–quantum communication latency, QPU programming, thermalization, and readout times. Tighter hardware integration can mitigate latency and programming delays; however, thermalization and readout overheads are expected to impact short annealing schedules more strongly. As a result, accounting for these effects could shift optimal strategies toward longer annealing times in practical settings.

As the QPU is typically invoked only once at the initialization stage for most problems, performance gains become more pronounced in cases where the grid configuration changes, such as in reconfiguration scenarios. While communication and queueing overheads may offset some gains, an adequately HPC-integrated QPU could enable deeper investigation and uncover new algorithmic performance benefits.

Lastly, we note that tailoring the optimization to the specific solver or software may further enhance performance and thus might be of interest for further studies.

References

[1] P. Denholm, D. J. Arent, S. F. Baldwin, D. E. Bilello, G. L. Brinkman, J. M. Cochran, W. J. Cole, B. Frew, V. Gevorgian, J. Heeter *et al.*, "The challenges of achieving a 100% renewable electricity system in the united states," *Joule*, vol. 5, no. 6, pp. 1331–1352, 2021.

[2] B. Badrzadeh, Z. Emin, E. Hillberg, D. Jacobson, L. Kocewiak, G. Lietz, F. da Silva, and M. Val Escudero, "The need or enhanced power system modelling techniques and simulation tools," *CIGRE Science & Engineering*, vol. 17, no. Febr, pp. 30–46, 2020.

[3] S. Subedi, M. Rauniyar, S. Ishaq, T. M. Hansen, R. Tonkoski, M. Shirazi, R. Wies, and P. Cicilio, "Review of methods to accelerate electromagnetic transient simulation of power systems," *IEEE Access*, vol. 9, pp. 89 714–89 731, 2021.

[4] J. Stürmer, A. Plietzsch, T. Vogt, F. Hellmann, J. Kurths, C. Otto, K. Frieler, and M. Anvari, "Increasing the resilience of the texas power grid against extreme storms by hardening critical lines," *Nature Energy*, vol. 9, no. 5, pp. 526–535, 2024.

[5] J. Tong, L. Hoang, N. Militello, C. Callaghan, Y. Chen, and N. Furtaw, "Speeding up dynamic security assessment," in *Proceedings of the IEEE Power and Energy Society General Meeting (PESGM)*. IEEE Power & Energy Society, 2023.

[6] M. Armstrong, J. Marti, L. Linares, and P. Kundur, "Multilevel MATE for efficient simultaneous solution of control systems and nonlinearities in the OVNI simulator," *IEEE Transactions on Power Systems*, vol. 21, no. 3, pp. 1250–1259, Aug. 2006.

[7] C. Hartmann, J. Zhang, C. D. Gonzalez Calaza, T. Pesch, K. Michielsen, and A. Benigni, "Quantum annealing based power grid partitioning for parallel simulation," *IEEE Transactions on Power Systems*, pp. 1–13, 2025.

[8] C. McGeoch and P. Farre, "The D-Wave Advantage System: An Overview," D-Wave, Tech. Rep., 2020.

[9] M. Mirz, S. Vogel, G. Reinke, and A. Monti, "DPsim—A dynamic phasor real-time simulator for power systems," *SoftwareX*, vol. 10, Jul. 2019.

[10] H. Happ, "Diakoptics—The solution of system problems by tearing," *Proceedings of the IEEE*, vol. 62, no. 7, pp. 930–940, Jul. 1974.

[11] J. Shu, W. Xue, and W. Zheng, "A Parallel Transient Stability Simulation for Power Systems," *IEEE Transactions on Power Systems*, vol. 20, no. 4, pp. 1709–1717, Nov. 2005.

[12] R. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. Miller and J. Thatcher, Eds. Plenum Press, 1972, pp. 85–103.

[13] A. Lucas, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. 2, 2014.

[14] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum Computation by Adiabatic Evolution," Jan. 2000.

[15] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model," *Physical Review E*, vol. 58, no. 5, pp. 5355–5363, Nov. 1998, publisher: American Physical Society.

[16] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Reviews of Modern Physics*, vol. 90, no. 1, p. 015002, Jan. 2018.

[17] D.-W. Systems, "Qpu timing and runtime limits," 2025, accessed: 2025-08-19. [Online]. Available: https://docs.dwavequantum.com/en/latest/quantum_research/operation_timing.html\#qpu-timing-runtime-limits

[18] C. D. Gonzalez Calaza, D. Willsch, and K. Michielsen, "Garden optimization problems for benchmarking quantum annealers," *Quantum Information Processing*, vol. 20, no. 9, p. 305, Sep. 2021.

[19] H. Munoz-Bauza and D. Lidar, "Scaling advantage in approximate optimization with quantum annealing," *Phys. Rev. Lett.*, vol. 134, p. 160601, Apr 2025.

[20] T. Sato, M. Ohzeki, and K. Tanaka, "Assessment of image generation by quantum annealer," *Scientific Reports*, vol. 11, no. 1, p. 13523, Jun. 2021.