Approximating under the Influence of Quantum Noise and Compute Power

Simon Thelen
Technical University of Applied Science Regensburg
Regensburg, Germany
simon.thelen@othr.de

Hila Safi
Siemens AG, Technology
Technical University of Applied Science Regensburg
Munich, Germany
hila.safi@siemens.com

Wolfgang Mauerer
Technical University of Applied Science Regensburg
Siemens AG, Technology
Regensburg/Munich, Germany
wolfgang.mauerer@othr.de

***Abstract*—The quantum approximate optimisation algorithm (QAOA) and its variants are at the core of many scenarios that aim at combining the power of quantum computers (QC) and classical high-performance computing (HPC) appliances for combinatorial optimisation. Several obstacles challenge concrete benefits now and in the foreseeable future: Imperfections quickly degrade algorithmic performance below practical utility; overheads arising from alternating between classical and quantum primitives can counter any advantage; and the choice of parameters or algorithmic variant can substantially influence runtime and result quality. Selecting the appropriate combination is a non-trivial issue, as it not only depends on end-user requirements, but also on details of the hardware and software stack.**

Appropriate automation can alleviate the burden of choosing optimal combinations for end-users: They should not be required to understand technicalities like detail differences between QAOA variants, required number of QAOA layers, or necessary measurement samples. Yet, they should receive the best possible satisfaction of their non-functional requirements, be it performance or other. We determine factors that influence approximation quality and temporal behaviour of four QAOA variants using comprehensive density-matrix-based numerical simulations targeting three widely studied optimisation problems. Our simulations consider ideal quantum computation, and a continuum of scenarios troubled by realistic imperfections.

Our quantitative results, accompanied by a comprehensive reproduction package, show strong performance differences between QAOA variants that can be pinpointed to narrow and specific effects. We identify influential co-variables and relevant non-functional quality goals that, as we argue, mark the relevant ingredients for designing appropriate software engineering abstraction mechanisms and automated tool-chains for devising quantum solutions from higher-level problem specifications.

***Index Terms*—QAOA, quantum noise, QC/HPC integration, design automation**

I. Introduction

Quantum computers have the potential to solve certain problems exponentially faster than classical computers. However, the capabilities of current noisy intermediate-scale quantum (NISQ) systems are limited by the influence of quantum noise that severely disturbs any quantum computation [1], and by the small number of available qubits, which inhibits quantum error correction [2] for now. Given the quantum-classical structure of NISQ-era algorithms, it is plausible that rather than replacing existing systems, quantum computers will be integrated into HPC (or embedded or special-purpose) architectures to accelerate specific tasks such as optimisation or simulation. This hybrid approach will allow HPC systems to leverage the strengths of both, quantum and classical computing, and will likely extend into the post-NISQ era.

Variational quantum algorithms such as the quantum approximate optimisation algorithm (QAOA) are at the forefront of current research [3]. QAOA uses a multi-layer, structured variational quantum circuit to find approximate solutions for a broad class of optimisation problems with many industrial applications. The more layers are used, the better the algorithm can approximate the optimal solution under ideal conditions. However, a deeper circuit on NISQ systems also increases the number of circuit parameters, execution time and, perhaps most importantly, susceptibility to noise. To complicate matters further, HPC engineers can choose between different QAOA variants such as warm-starting QAOA (WSQAOA) [4] or recursive QAOA (RQAOA) [5, 6]. These address noise by putting a stronger focus on the classical part of the algorithm.

Depending on the specific use-case, different non-functional requirements such as required solution quality, execution time or noise resistance must be taken into account. This results in a number of potential trade-offs that must be carefully considered to put QAOA to good use in HPC environments. It is not obvious a-priori which variant performs best for which task, given the constraints of NISQ hardware and the desired result quality. Ideally, selecting a suitable QAOA variant should be done automatically by the compiler or runtime environment. However, the effects of noise on QAOA are not fully understood, especially for non-standard variants of the algorithm. To the best of our knowledge, no comprehensive study compares different QAOA variants under noise.

Our work, which is accompanied by a code repository and a reproduction package (links in PDF), aims to address this knowledge gap by examining the effect of factors like QAOA variant, number of layers, and noise strength on the non-functional properties of programs in HPC systems augmented with NISQ hardware. We use density-matrix-based numerical simulations to analyse solution quality and execution times of four QAOA variants on three widely studied optimisation problems. Our objective is to identify factors with the strongest

influence on non-functional quality goals, to guide the design and development of appropriate abstraction mechanisms and automatic tools based on high-level problem descriptions.

II. Context and Foundation

A. Subject Problems

We study the performance of multiple QAOA variants using three NP-complete optimisation problems: Max-Cut (which seeks to partition the vertices of an undirected graph into two disjoint sets S and T such that the number of edges between S and T is minimised), Partition (which seeks to distribute a set of numbers into disjoint subsets S and T such that the absolute difference between the sum of numbers in S and T is minimised) and Vertex Cover (which seeks to find the smallest vertex subset C of an undirected graph such that for every edge (u, v) in the graph, either $u \in C$ or $v \in C$). The subject problems (a) are well-understood, with many applications, (b) have efficient encodings with only one qubit per vertex (Max-Cut, Vertex Cover) or number (Partition), and (c) differ considerably in their hardness of approximation: A fully polynomial-time approximation scheme is known for Partition [7], but the best-known approximation ratios are 0.878 for Max-Cut [8] and 2 for Vertex Cover [9].

B. QAOA and its variants

QAOA finds approximate solutions to combinatorial optimisation problems [3]. The basic algorithm seeks parameters s to minimise $C(s) = -\sum_{i<j} J_{ij} s_i s_j - \sum_i h_i s_i$ of a quadratic binary unconstrained objective function (QUBO) with $s_i = \pm 1$ and $J_{ij}, h_i \in \mathbb{R}$. Many NP-complete optimisation problems, including our subject problems, have efficient QUBO encodings [10]. A QAOA circuit first prepares an initial state, typically $|+\rangle^n$, and applies a series of unitaries: $e^{-\mathrm{i}\beta_p H_M} e^{-\mathrm{i}\gamma_p H_C} \dots e^{-\mathrm{i}\beta_1 H_M} e^{-\mathrm{i}\gamma_1 H_C}$, where p chooses the number of *layers*. H_C is the *problem Hamiltonian* (or separator, as it separates solution space from search space by a complex phase), with $H_C|x\rangle = C((-1)^{x_1}, \dots, (-1)^{x_n})|x\rangle$. $H_M = \sum_i X^{(i)}$ is called the *mixer Hamiltonian*, which influences the quantum state to explore the search space. Optimal parameters β_i, γ_i $(1 \le i \le p)$ are found through multiple circuit evaluations using a classical optimiser. More layers improve, in principle, results at the expense of runtime, but also amplify NISQ noise, which decreases solution quality.

Warm-starting QAOA variants alter the initial state and/or mixer Hamiltonian using a known initial guess for the solution. This can improve result quality, especially at low depths [4]. We consider two variants: For WS-Init-QAOA, given an approximate solution $z^* \in \{-1, +1\}^n$, we use $\bigotimes_i RY(\theta_i)|0\rangle$ with $\theta_i = 2 \arcsin\left(\sqrt{0.5 - 0.25 z_i^*}\right)$ as the initial state instead of $|+\rangle^n$ [11]. For WSQAOA, in addition to the altered initial state, we also change the time-evolved mixer Hamiltonian to $\bigotimes_i RY(-\theta_i) RZ(-2\beta) RY(\theta_i)$ [4].

Recursive QAOA (RQAOA) [5, 6] samples the circuit with the final parameters $\vec{\beta}$ and $\vec{\gamma}$ after QAOA execution. Then, the individual terms of the Hamiltonian $T = \{s_i : h_i \neq 0\} \cup \{s_i s_j : J_{ij} \neq 0\}$ are considered: Term $t \in T$ with the largest absolute expected value $|E[t]|$ according to sampling results is selected, and constraint $t = \mathrm{sign}(E[t])$ is inserted into the Hamiltonian by substituting $s_i = \pm 1$ or $s_i = \pm s_j$, which eliminates one variable. This process of using QAOA to find the "most conclusive" term is iterated until the problem becomes trivial to solve. Finally, a solution that satisfies all encountered constraints is selected.

III. Related Work

QAOA [3] is among the most prevalent algorithms considered for NISQ devices. A wide range of efforts focuses on optimising QAOA for available quantum hardware, particularly to mitigate the effect of high error rates. Modifications to both, structure of the quantum circuit (*e.g.*, Refs. [5, 12–15]) and classical optimisation procedure (*e.g.*, Refs. [4, 11, 16–20]) have been proposed. We cannot consider the abundance of choices, but focus on typical variants, as described above: WSQAOA [4], WS-Init-QAOA [11] and RQAOA [5].

As for quantum noise, Georgopoulos *et al.* [21] present an approach to simulate effects of three error types using quantum channels, and align the model with experimental observations. Greiwe *et al.* [22] investigate the effects of imperfections on quantum algorithms. Xue *et al.* [23] confirm the effectiveness of hybrid algorithms on NISQ devices by studying effects of quantum noise on standard QAOA. Marshall *et al.* [24] provide an approximate model for fidelity and expected cost given noise rate, system size, and circuit depth.

Integrating QC in HPC environments poses challenges, and comprehensive software stacks to address these are being designed and implemented [25–31]; alas, we cannot review all in detail. Bandic *et al.* [28] survey QC full-stacks, highlighting the need for tight co-design and vertical integration between software and hardware. Auto-tuning in HPC environments was studied by Hoefler *et al.* [32]. Wintersperger *et al.* [33] and Safi *et al.* [34] study the influence of parameters like communication latencies and adapted topologies in HPC/QC systems. Elsharkawy *et al.* [30] assess the suitability of quantum programming tools for integration with classical HPC frameworks. Close integration of classical and quantum aspects is a paramount desire in most studies.

IV. Experimental Setup

We compare the ideal and noisy performance of different QAOA variants using density matrix-based, numerical simulations. We use the Eviden Qaptiva 800 quantum simulation platform and its proprietary software library *QLM* that includes a high-performance noisy circuit simulator. Different numbers of layers ($p \in \{1, 2, 3, 4\}$) are investigated for each variant. We will analyse performance on random instances of the three problems *Max-Cut*, *Partition* and *Vertex Cover*: For Max-Cut and Vertex Cover, 600 random graphs are considered, 100 for each $n \in \{5, 6, 7, 8, 9, 10\}$. These graphs are created by inserting an edge between every pair of vertices with probability $p = 0.5$. For Partition, 100 sets of numbers are considered for every size $n \in \{5, 6, 7, 8, 9, 10\}$. Each number

is drawn uniformly at random from the interval $[0, 1]$. To obtain more accurate data, the algorithms are run multiple times on each instance, averaging the results. For all variants, SciPy's COBYLA optimiser with a tolerance of 1 % and 150 maximum iterations is used as the classical parameter optimiser [35, 36]. The warm-starting variants use approximate solutions obtained from the Goemans-Williamson algorithm for Max-Cut [8], greedy list scheduling for Partition [37] and the classic two-approximation algorithm vor Vertex Cover [38].

To compare algorithm performance, we consider *approximation quality*, here defined as the ratio between the obtained solution and the optimal solution for a problem-specific measure (Max-Cut: size of the cut; Partition: cardinality of the smaller set; Vertex Cover: reciprocal number of vertices in the cover[1]). A approximation quality of 1 corresponds to the optimal, and 0 to the worst possible solution.

Using density-matrix-based simulations is computationally more expensive than other alternatives, but allows us to obtain *exact* values for $\langle\psi|H_C|\psi\rangle$—the expected energy of the problem Hamiltonian—in noisy simulations. For QAOA, WSQAOA and WS-Init-QAOA, we obtain the average value (and average approximation quality) of the solution from the measurement probabilities of the output state. For RQAOA, this approach is not possible since the algorithm delivers only a single outcome per design, which we need to appropriately account for. During RQAOA, when using a reasonably small sample size, we cannot expect to always find the term t with maximum $|E[t]|$. Therefore, we do not use the exact measurement probabilities to find the most "conclusive" term. Instead, we consider a constant sample size of ten. As shown below, this still achieves good results.

A. Noise Model

The Qiskit noise model is widely used [21, 22, 39], and accounts for dominant noise in IBM quantum systems, featuring good agreement with actual quantum hardware [21, 40]. We implemented this model on top of QLM's noisy circuit simulator, but additionally allow for changing the strength of the individual noise sources. Quantum channels describe

- *Gate errors:* Quantum gates can never be implemented perfectly. This manifests itself in the fact that instead of the desired unitary, a different, unknown unitary or non-unitary operator is applied instead. Gate errors are modelled using depolarising channels, which with some depolarising probability p replace the state of the affected qubits by the maximally mixed state [41].
- *Thermal relaxation:* Even if the qubit is not involved in any quantum gates, its state slowly transitions to the thermal equilibrium state $|0\rangle$ over time [41, 42].

The model is parameterised by (a) longitudinal (T_1) and transverse (T_2) relaxation time, (b) gate error and duration for each gate type (*e.g.*, R_Z or C–X), (c) strength of gate errors (d_D) and thermal relaxation (d_{TR}) as discussed in Section IV-B. As a simplifying assumption, we assume all qubits share identical imperfection characteristics.

	R_Z	$\sqrt{X}$	C–X	T_1	T_2
Gate error	0 %	0.03 %	1 %		
Duration | Time	0 ns	35 ns	400 ns	100 µs	150 µs

TABLE I: Baseline noise parameters.

Table I shows our baseline parameters: They represent median values obtained from the 46 IBM QPUs whose data are available from fake backends. For each backend, we take the median over all qubits, and then obtain the median over backend medians to ensure that systems with more qubits are not disproportionately represented. T_1 and T_2 are rounded to 5 µs, gate times to 5 ns and gate errors to one significant digit. These values represent the current state of the art in transmonic NISQ devices. For lack of space, we cannot include other architectures, but our simulations can be easily performed for them using the reproduction package [43] (link in PDF).

The *average fidelity* of a quantum channel measures how much it alters its input. It lies in the interval $[2^{-n}, 1]$ where n is the number of qubits the channel acts upon. An average fidelity of 1 means that the channel leaves its input unchanged, an average fidelity of 2^{-n} represents a complete loss of information. Our model interprets the empirically determined gate error as one minus its average fidelity. We assume this infidelity to be caused by both error types, which we model as depolarising noise and thermal relaxation.

After each logical single-qubit gate, we insert a thermal relaxation channel, parameterised by T_1, T_2 and gate duration t, followed by a depolarising channel. The depolarising probability p is set such that the average fidelity of the composed channel matches the model parameters (the required probability can be derived analytically [22, 44]). After each two-qubit gate, one thermal relaxation channel is inserted for each involved qubit, followed by a two-qubit depolarising channel (again, we choose parameters so that the combined channel matches the desired fidelity). Due to two-qubit gate dependencies, some qubits might not be subject to any gate during *idle periods*. These are represented by a thermal relaxation channel parameterised by T_1, T_2, and idle duration.

We assume a native gate set $\{R_Z, \sqrt{X}, \text{C–}X\}$, as it is supported by many transmon devices, and transpile from logical to native circuits by substituting single-qubit gates. While not all NISQ machines support C–X gates natively (*e.g.*, the IBM Eagle), equivalents (up to single-qubit rotations) exist on all architectures. Since two-qubit gate imperfections usually exceeds single-qubit gate imperfections, any additional single-qubit gates that arise from substituting C–X should not meaningfully affect our results. R_Z gates are noiseless, as they are virtual on transmon systems [45], and Misra-Gries edge colouring [46] orders terms in H_C to minimise circuit depth.

[1]Invalid Vertex Cover solutions are replaced with the trivial vertex cover using the entire vertex set.

B. Strength of Noise

Recall that noise strengths are controlled via parameters d_D and d_{TR}. Specifically, d_D scales the probability p of all depolarising channels, d_{TR} scales the time parameter t of all thermal relaxation channels. A value of 1 corresponds to the baseline noise level, whereas a value of 0 disables the noise source. Intuitively, depolarising and thermal relaxation noise lead to a decrease of the state's quality (as measured by fidelity with the desired optimal state), which is exponential in the circuit's depth. The average fidelity of the depolarising channel is $1/2 + 1/2 \cdot e^{-t/T_1}$ if $T_1 = T_2$, and d_D is a scaling factor of the decay constant $(1/T_1)$ [41, 44]. Likewise, the average fidelity after k-fold application of the single-qubit depolarising channel with depolarisation probability p is $1/2 + 1/2 \cdot e^{k \ln(1-p)}$ [47]. As $\ln(1-p) \approx p$ for $p \ll 1$, d_D again scales the decay constant for small p.

C. Runtime estimation

To estimate circuit execution time, we use gate durations given in Table I, and a median measurement time of 4.09 µs obtained from fake backends. We assume 1000 circuit evaluations per iteration. Measurements on the Qaptiva 800 determine efforts for parameter optimisation, circuit transpilation, or finding initial values for warm-starting variants.

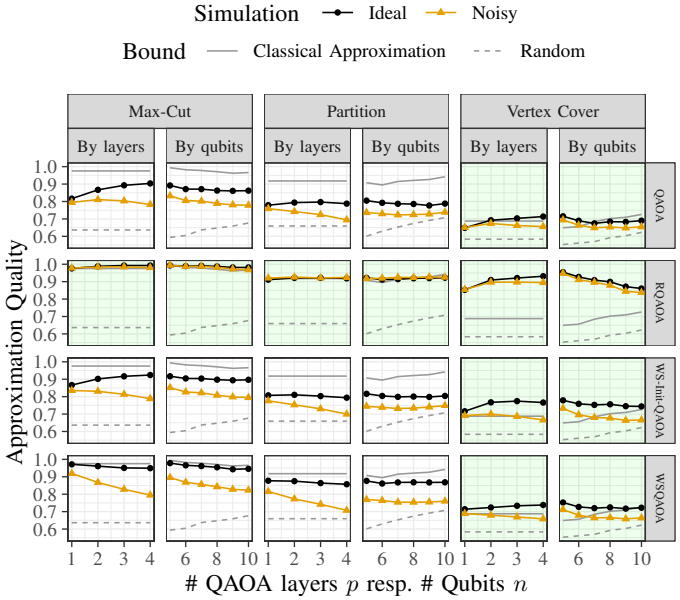

Fig. 1: Average approximation quality, separated by number of QAOA layers and averaged over number of qubits and vice versa, on random problem instances, for ideal and noisy evaluation. Dashed grey lines: lower bound (random guess). Solid grey lines: initial warm-start estimate. Green background: QAOA variant outperforms classical approximation algorithm.

V. Evaluation

Our core concern is to understand how non-functional properties of QAOA variants affect their integration into HPC systems, with a particular focus on the effect of imperfections on solution quality and runtime. Figure 1 summarises our general observations: It shows average approximation quality, dependent on an increasing number of layers p and averaged over the problem size n, as well as depending on n and averaged over p, for two corner cases: Full IBM Q device noise, and the noiseless case. Overall, RQAOA performs best and standard QAOA worst in the ideal and the noisy cases. WSQAOA and WS-Init-QAOA ranked second and third, respectively. Although, as expected, all variants perform worse in the noisy case, there is a clear difference in performance deterioration; particularly, RQAOA is more robust than other variants. For an increasing number of layers, the better result quality in the noiseless case competes with a larger amount of sustained noise. For IBM Q devices, detrimental effects usually exceed improvements, and using two QAOA layers instead of one shows only an improvement for standard QAOA and RQAOA. For the two warm-starting variants, the added power of a second layer does not outweigh the additional noise. While WSQAOA is slightly more affected, the effect of noise is mostly comparable across the other variants.

It appears reasonable that RQAOA is less disturbed by noise: Instead of trying to approximate an optimal solution directly, RQAOA iteratively finds the most "conclusive" constraint and adds it to the Hamiltonian. This way, it finds the most highly correlated pair of variables in every step, which are then fixed and not affected by future influence of noise. WSQAOA struggles to benefit from adding more layers, even in the ideal case, which we attribute to the limitations of the relatively "weak" classical optimiser.

Figure 2 compares solution quality and runtime for IBM Q noise levels. Each data point represents average approximation quality and runtime of an algorithm with $p \in [1, 4]$ layers for 100 problem instances with the same number of qubits. We observe that (a) RQAOA is consistently best in terms of solution quality, albeit at the expense of runtime, since it executes QAOA multiple times, (b) differences in intra-algorithm execution times are mostly negligible for non-RQAOA variants, (c) approximation quality decreases, for increasing problem dimension, at markedly different rates depending on the subject problem, and specific to each algorithm, (d) said decrease follows, from visual observation, a characteristic trend, (e) distinct clusters arise that are connected to the amount of layers. Assuming this behaviour remains valid for larger instances (which is, of course, not guaranteed [48]), the clustering presents an opportunity to select suitable execution parameters prior to payload execution.

Finally, we consider the continuum between the noise-free case and IBM Q imperfection levels: Figure 3 shows under what conditions adding an extra layer improves performance. We use relative advantage (*i.e.*, the ratio of p layer performance over $p-1$ layer performance) averaged over all subject problem instances. For moderate noise, standard QAOA benefits from adding a second or third layer, as does WS-Init-QAOA when adding a second. However, this depends on the subject problem: Partition is less resistant to noise than Max-Cut and

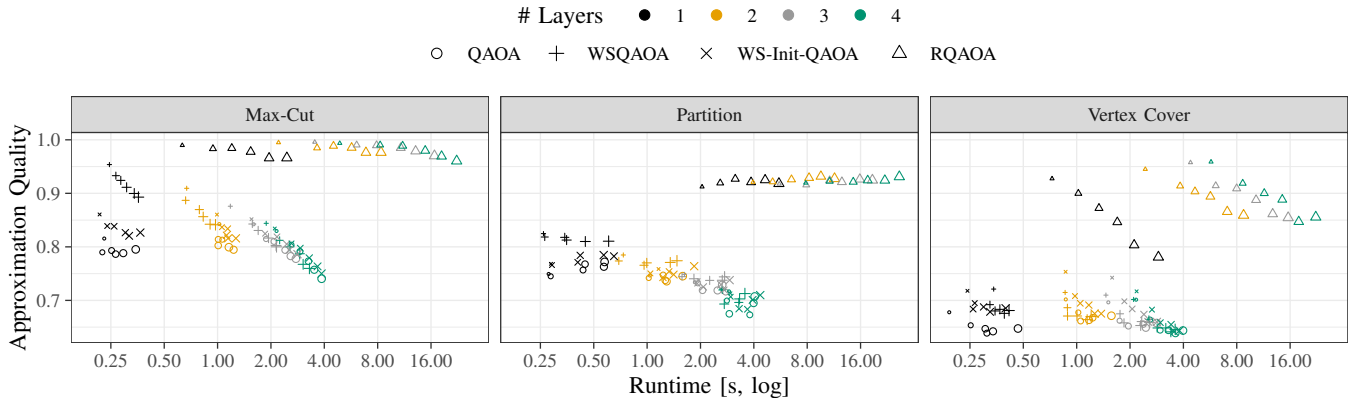

Fig. 2: Runtime versus approximation quality for noisy simulation. Each point represents an average over 100 instances for a given number of qubits; point size increases with instance size respectively qubit count (five to ten).

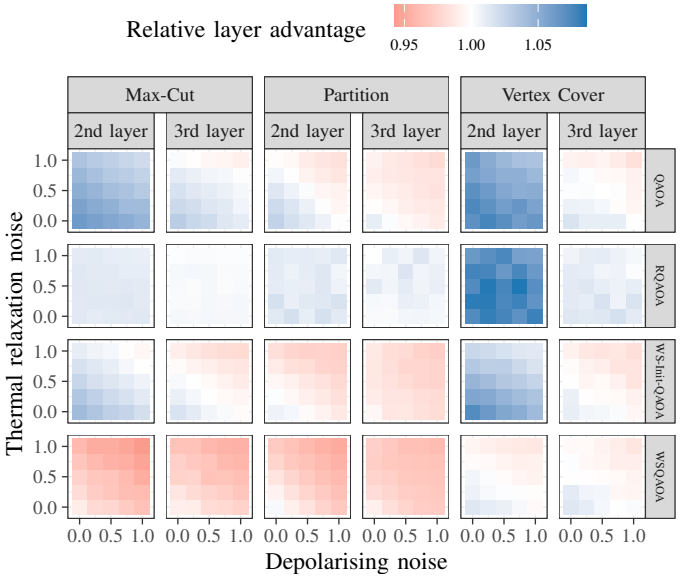

Fig. 3: Relative advantage of a second and third QAOA layer depending on noise strength. Blue cells: performance increase; red cells: worse performance with added layer(s).

Vertex Cover. A likely reason is that circuits for Partition contain more C–X gates than those for Max-Cut or Vertex Cover instances: For Partition, there is one C–X gate for every pair of qubits; for Max-Cut and Vertex Cover there is one C–X for each graph edge, which amounts to roughly half the pairs, since for each pair, an edge is inserted with 50 % probability.

It is notable that in the majority of cases, relative advantage is consistently *either* negative *or* positive for a given combination of algorithm and problem. A change in trend depending on noise strength is rare. This suggests that choosing the optimal number of layers does not need to take into account hardware noise details.

VI. Discussion & Conclusion

Our results show pronounced differences between the QAOA variants for solution quality, runtime or noise strength that can be pinpointed to specific causes. This suggests that augmenting software stacks with capabilities to automatically select optimal QAOA variants with an appropriate number of layers for a given problem, based on non-functional criteria such as solution quality, latency, or quantum runtime is a possible and useful endeavour. Our data suggest that in noisy environments, RQAOA has the potential to achieve better results than other variants, at the cost of a much longer runtime. They also indicate that for most QAOA variants, the computational benefit of a second layer is often greatly reduced or even eliminated by the added noise of the deeper quantum circuits. The considerable computational cost of simulating noisy quantum circuits allowed us to only consider problems with up to ten qubits. Nonetheless, we believe that results for relatively small instances may indicate general trends, although this needs further confirmation. If our expectation hold true, another possible approach to (semi-)automatic algorithm selection could be to probe smaller dummy instances at compile time or runtime to inform the selection for real problem instances.

Automatic algorithm selection needs to distinguish between problem-independent and problem-specific elements, and abstractions for deploying quantum algorithms such as QAOA in HPC systems need to take this into account. While the trade-off between solution quality and runtime remains relatively consistent across the three subject problems when considering the algorithm and noise dimensions, the situation becomes more nuanced when examining different numbers of QAOA layers. The point at which an additional layer does not yield performance improvements depends on the problem itself and on other factors like noise level or problem size. Other aspects—for instance, the impact of problem encodings [49, 50] or noise caused by swap gates required because of limited connectivity [34]—, should also be considered. More research beyond our results is needed to understand the effect of problem-specific aspects such as classical approximability on the optimal hyper-parameters of different QAOA variants.

To enjoy QC in HPC environments without the need for quantum training, the use of QAOA (and possibly other classes

of algorithms) could be enabled by allowing users to specify relevant non-functional requirements (*e.g.*, acceptable solution quality or communication latency) in quantum code [51–53], for instance via meta-annotations. If augmented with appropriate (and already available [54]) means of providing objective functions decoupled from quantum specifics, yet convertible to QUBO, and together with apt classical means of providing initial solution estimates, this would allow a compiler or runtime system to select the most suitable algorithm. Our results suggest that this vision is achievable in the near future.

Acknowledgements: This work was supported by the German Federal Ministry of Education and Research (BMBF), within the funding program "quantum technologies—from basic research to market", grant numbers 13NI6092 (WM, ST) and 13N16093 (HS). WM and ST acknowledge support by the High-Tech Agenda of the Free State of Bavaria.

References

[1] J. Preskill, *Quantum computing in the NISQ era and beyond*, *Quantum*, vol. 2, p. 79, 2018.

[2] J. Roffe, *Quantum error correction: An introductory guide*, *Contemporary Physics*, vol. 60, no. 3, pp. 226–245, 2019.

[3] E. Farhi, J. Goldstone, and S. Gutmann, *A quantum approximate optimization algorithm*, 2014. arXiv: 1411.4028 [quant-ph].

[4] D. J. Egger, J. Mareček, and S. Woerner, *Warm-starting quantum optimization*, *Quantum*, vol. 5, p. 479, 2021.

[5] S. Bravyi *et al.*, *Obstacles to variational quantum optimization from symmetry protection*, *Phys. Rev. Lett.*, vol. 125, 2020.

[6] E. Bae and S. Lee, *Recursive QAOA outperforms the original QAOA for the MAX-CUT problem on complete graphs*, *Quantum Information Processing*, vol. 23, no. 3, p. 78, 2024.

[7] O. H. Ibarra and C. E. Kim, *Fast approximation algorithms for the knapsack and sum of subset problems*, *J. ACM*, vol. 22, no. 4, 1975.

[8] M. X. Goemans and D. P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, *J. ACM*, vol. 42, no. 6, pp. 1115–1145, 1995.

[9] G. Karakostas, *A better approximation ratio for the vertex cover problem*, *ACM Trans. Algorithms*, vol. 5, no. 4, 2009.

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

[11] A. Awasthi *et al.*, *Quantum computing techniques for multi-knapsack problems*, in *Intelligent Computing*, Springer Nature, 2023.

[12] Z. Jiang, E. G. Rieffel, and Z. Wang, *Near-optimal quantum circuit for grover's unstructured search using a transverse field*, *Phys. Rev. A*, vol. 95, 6 2017.

[13] Z. Wang *et al.*, *XY Mixers: Analytical and numerical results for the quantum alternating operator ansatz*, *Phys. Rev. A*, vol. 101, 1 2020.

[14] A. Bärtschi and S. Eidenbenz, *Grover Mixers for QAOA: Shifting complexity from mixer design to state preparation*, in *Proc. IEEE QCE*, 2020, pp. 72–82.

[15] L. Zhu *et al.*, *Adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer*, *Phys. Rev. Res.*, vol. 4, 2022.

[16] R. Tate *et al.*, *Warm-Started QAOA with Custom Mixers Provably Converges and Computationally Beats Goemans-Williamson's Max-Cut at Low Circuit Depths*, *Quantum*, vol. 7, p. 1121, 2023.

[17] V. Vijendran *et al.*, *An expressive ansatz for low-depth quantum approximate optimisation*, *Quantum Sci. and Tech.*, vol. 9, 2024.

[18] J. Sud *et al.*, *Parameter-setting heuristic for the quantum alternating operator ansatz*, *Phys. Rev. Res.*, vol. 6, 2024.

[19] J. A. Montanez-Barrera, D. Willsch, and K. Michielsen, *Transfer learning of optimal qaoa parameters in combinatorial optimization*, 2024. eprint: 2402.05549.

[20] M. Streif and M. Leib, *Training the quantum approximate optimization algorithm without access to a quantum processing unit*, *Quantum Science and Technology*, vol. 5, 2020.

[21] K. Georgopoulos, C. Emary, and P. Zuliani, *Modeling and simulating the noisy behavior of near-term quantum computers*, *PRA*, 2021.

[22] F. Greiwe, T. Krüger, and W. Mauerer, *Effects of imperfections on quantum algorithms: A software engineering perspective*, in *Proc. IEEE International Conference on Quantum Software*, 2023.

[23] C. Xue *et al.*, *Effects of quantum noise on quantum approximate optimization algorithm*, *Chinese Physics Letters*, vol. 38, 2021.

[24] J. Marshall *et al.*, *Characterizing local noise in QAOA circuits*, *IOP SciNotes*, vol. 1, no. 2, 2020.

[25] T. S. Humble *et al.*, *Quantum computers for high-performance computing*, *IEEE Micro*, vol. 41, no. 5, pp. 15–23, 2021.

[26] Y. Alexeev *et al.*, *Quantum-centric supercomputing for materials science: A perspective on challenges and future directions*, *Future Generation Computer Systems*, vol. 160, pp. 666–710, 2024.

[27] P. J. Karalekas *et al.*, *A quantum-classical cloud platform optimized for variational hybrid algorithms*, *Q. Sci. & Tech.*, vol. 5, no. 2, 2020.

[28] M. Bandic, S. Feld, and C. G. Almudever, *Full-stack quantum computing systems in the NISQ era: Algorithm-driven and hardware-aware compilation techniques*, in *Proc. DAAD*, 2022, pp. 1–6.

[29] M. N. Farooqi and M. Ruefenacht, *Exploring hybrid classical-quantum compute systems through simulation*, in *IEEE QCE*, vol. 02, 2023, pp. 127–133.

[30] A. Elsharkawy *et al.*, *Integration of Quantum Accelerators with High Performance Computing – A Review of Quantum Programming Tools*, 2023.

[31] C. Campbell *et al.*, *Superstaq: Deep optimization of quantum programs*, in *IEEE QCE*, 2023, pp. 1020–1032.

[32] T. Hoefler and R. Belli, *Scientific benchmarking of parallel computing systems: Twelve ways to tell the masses when reporting performance results*, in *Proc. Int. Conf. for HPC, Networking, Storage and Analysis*, ACM, 2015.

[33] K. Wintersperger, H. Safi, and W. Mauerer, *QPU-System Co-design for Quantum HPC Accelerators*, in *Proc. ARCS*, 2022, pp. 100–114.

[34] H. Safi, K. Wintersperger, and W. Mauerer, *Influence of hw-sw-co-design on quantum computing scalability*, in *IEEE International Conference on Quantum Software*, 2023, pp. 104–115.

[35] P. Virtanen *et al.*, *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*, *Nature Methods*, vol. 17, pp. 261–272, 2020.

[36] M. J. D. Powell, *A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation*, in *Advances in Optimization and Numerical Analysis*, Springer, 1994.

[37] R. L. Graham, *Bounds for certain multiprocessing anomalies*, *The Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

[38] T. H. Cormen *et al.*, *Introduction to algorithms*, Fourth edition. Cambridge, Massachusett: The MIT Press, 2022.

[39] C. Blank *et al.*, *Quantum classifier with tailored quantum kernel*, *npj Quantum Information*, vol. 6, no. 1, p. 41, 2020.

[40] T. Weber *et al.*, *Construction and volumetric benchmarking of quantum computing noise models*, *Physica Scripta*, vol. 99, no. 6, 2024.

[41] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.

[42] P. Krantz *et al.*, *A quantum engineer's guide to superconducting qubits*, *Applied Physics Reviews*, vol. 6, no. 2, 2019.

[43] W. Mauerer and S. Scherzinger, *1-2-3 reproducibility for quantum software experiments*, *IEEE SANER*, pp. 1247–1248, 2022.

[44] M. A. Nielsen, *A simple formula for the average gate fidelity of a quantum dynamical operation*, *Physics Letters A*, vol. 303, 2002.

[45] D. C. McKay *et al.*, *Efficient Z-Gates for Quantum Computing*, *Physical Review A*, vol. 96, no. 2, 2017.

[46] J. Misra and D. Gries, *A constructive proof of vizing's theorem*, *Information Processing Letters*, vol. 41, no. 3, pp. 131–133, 1992.

[47] E. Magesan, *Depolarizing behavior of quantum channels in higher dimensions*, *Quantum Info. Comput.*, vol. 11, no. 5, 2011.

[48] C. C. McGeoch and P. Farré, *Milestones on the quantum utility highway: Quantum annealing case study*, *ACM TQC*, vol. 5, 2023.

[49] T. Krüger and W. Mauerer, *Quantum annealing-based software components: An experimental case study with sat solving*, in *Proc. ICSEW*, New York, NY, USA: ACM, 2020, pp. 445–450.

[50] S. Zielinski *et al.*, *SATQUBOLIB: A python framework for creating and benchmarking (Max-)3SAT QUBOs*, in *Proc. I4CS*, Springer, 2024, pp. 48–66.

[51] M. Schönberger *et al.*, *Peel | pile? cross-framework portability of quantum software*, in *IEEE ICSA-C*, 2022, pp. 164–169.

[52] W. Mauerer, *Semantics and simulation of communication in quantum programming*, *arXiv preprint quant-ph/0511145*, 2005.

[53] J. Lee *et al.*, *Quantum task offloading with the openmp api*, in *SC23*, 2023.

[54] E. Lobe, *Quark: Quantum application reformulation kernel*, in *INFORMATIK 2023*, Bonn: GI e.V., 2023, pp. 1115–1120.