Scalable Quantum Optimisation using HADOF: Hamiltonian Auto-Decomposition Optimisation Framework

Namasi G. Sankar[1,2,*], Georgios Miliotis[3] and Simon Caton[1,2]

[1]School of Computer Science, University College Dublin, Ireland
[2]Centre for Quantum Engineering, Science, and Technology, University College Dublin, Ireland
[3]Antimicrobial Resistance and Microbial Ecology Group, School of Medicine, University of Galway, Galway, Ireland

Abstract

Quantum Annealing (QA) and QAOA are promising quantum optimisation algorithms used for finding approximate solutions to combinatorial problems on near-term NISQ systems. Many NP-hard problems can be reformulated as Quadratic Unconstrained Binary Optimization (QUBO), which maps naturally onto quantum Hamiltonians. However, the limited qubit counts of current NISQ devices restrict practical deployment of such algorithms. In this study, we present the Hamiltonian Auto-Decomposition Optimisation Framework (HADOF), which leverages an iterative strategy to automatically divide the Quadratic Unconstrained Binary Optimisation (QUBO) Hamiltonian into sub-Hamiltonians which can be optimised separately using Hamiltonian based optimisers such as QAOA, QA or Simulated Annealing (SA) and aggregated into a global solution. We compare HADOF-with Simulated Annealing (SA) and the CPLEX exact solver, showing scalability to problem sizes far exceeding available qubits while maintaining competitive accuracy and runtime.. Furthermore, we realize HADOF for a toy problem on an IBM quantum computer, showing promise for practical applications of quantum optimisation.

Keywords

Scalable Quantum Optimisation, Quantum Approximate Optimisation Algorithm (QAOA), Quantum Annealing (QA), Simulated Annealing (SA), Cplex, Divide and Conquer

1. Introduction

The Quadratic Unconstrained Binary Optimisation (QUBO) model provides a unified framework for formulating many combinatorial optimization problems—such as the Travelling Salesman Problem (TSP), graph partitioning, and scheduling—which are often NP-hard and difficult to scale using classical exact solvers [10]. While specialized heuristics exist (e.g., Lin–Kernighan for TSP) [16], they lack generality. In contrast, general-purpose QUBO solvers, including IBM Cplex [4] and other MILP/QP engines [1], offer flexibility but struggle with large instances [10]. QUBO also has the advantage of being represented as a Hamiltonian naturally, which can then be optimised via quantum computing [11] and classical Simulated Annealing (SA) [9].

Quantum algorithms theoretically provide a scaling advantage for certain optimisation problems over classical methods [1, 15]. Some quantum QUBO algorithms include Quantum Approximate Optimisation Algorithm (QAOA) [5] , Quantum Annealing (QA) [18] and Grover Adaptive Search (GAS) [6]. QAOA and QA provide multiple approximately optimal solutions in parallel, by taking advantage of quantum superposition. This is useful for many applications as it allows the domain expert to choose the best fitting solution for their particular problem and also compare different solutions.

3rd International Workshop on AI for Quantum and Quantum for AI (AIQxQIA 2025), at the 28th European Conference on Artificial Intelligence (ECAI), October 25-30, 2025, Bologna, Italy
*Corresponding author.
namasivayam.gomathisankar@ucdconnect.ie (N. G. Sankar); georgios.miliotis@universityofgalway.ie (G. Miliotis); simon.caton@ucd.ie (S. Caton)
0009-0000-7787-7490 (N. G. Sankar); 0000-0002-0944-2206 (G. Miliotis); 0000-0001-9379-3879 (S. Caton)

[1]GNU Linear Programming Kit, available at: http://www.gnu.org/software/glpk/glpk.html, last visited: January 7, 2026

However, current quantum devices in the NISQ era have a limited number of qubits and cannot (yet) be used for practical and scalable applications [6]. In this study, we propose the Hamiltonian Auto-Decomposition Optimisation Framework (HADOF), a framework for the automatic decomposition of a global Hamiltonian into sub-Hamiltonians, using an iterative optimisation process. The HADOF framework can be used to scale up many QUBO based algorithms such as QAOA, QA, Feedback-Based Quantum Optimisation (FALQON) [13] and SA [18]. Algorithms which produce a probability distribution over the solution space, from which solutions can be sampled - where good solutions are more likely to be sampled (approximately) are compatible with HADOF.

HADOF recovers more information from the sampling distribution, beyond merely the single best solution, enabling HADOF to scale to problem sizes much larger than the available number of qubits. We demonstrate, through classical simulation of QAOA within our framework, that HADOF surpasses Cplex on QUBO instances out of reach under the same classical hardware conditions, producing multiple high-quality solutions concurrently. Moreover, we argue that on actual quantum hardware, HADOF would exhibit even greater performance acceleration, combining quantum advantage with heuristic flexibility. Our results show promise for HADOF both as a quantum-inspired classical algorithm and as a scalable method on NISQ-era and future quantum devices.

2. Related Work

QAOA [5] and QA [18] are foundational quantum methods for tackling QUBO problems. QAOA is a variational, gate-based algorithm alternating between cost and mixer Hamiltonians, generating a probability distribution over solutions favoring low-cost solutions [11, 8]. QA, on the other hand, is an analog adiabatic method evolving a quantum system from an initial to the problem Hamiltonian. Both produce biased sample distributions over candidate solutions, offering practitioners flexibility when selecting among high-quality alternatives. However, NISQ hardware limits QAOA/QA to small problem sizes due to qubit and connectivity constraints [8]. This motivates hybrid and decomposition approaches.

In classical optimization, divide-and-conquer and decomposition heuristics are standard for scaling to large problems. General purpose solvers (e.g., IBM Cplex [4]) can struggle QUBO problems beyond hundreds of variables [10], motivating decomposition and hybrid quantum-classical methods. The multilevel QAOA of Maciejewski *et al.* [12] splits a large QUBO into manageable sub-QUBOs that are solved iteratively or in parallel and then recombined. These techniques enable practical scaling and lay the foundation for distributed quantum optimization.

Recent strategies distribute or decompose QAOA across subproblems. Recursive QAOA (RQAOA) [3] uses QAOA to iteratively fix qubits and shrink the problem, focusing quantum resources on the hardest sub-instances. The QAOA-in-QAOA ($QAOA^2$) and related parallel QAOA heuristics [19] decompose a large graph (e.g., MaxCut) into subgraphs, solve each with QAOA in parallel, and merge the results, exploiting high-performance computing (HPC) for scalability. Early approaches worked best on sparse or weakly coupled problems [8], but dense QUBOs require advanced coordination to manage strong variable interactions.

The Distributed QAOA (DQAOA) framework [8] extends parallelization further. Large QUBOs are decomposed into sub-QUBOs, solved on quantum or classical resources in parallel, with an aggregation policy reconciling overlaps and correlated interactions. This iterative approach scales to large, dense QUBOs; for example, Kim et al. report $\tilde{9}9\%$ approximation ratios on 1,000-variable instances within minutes, outperforming prior methods in both quality and time-to-solutionon. DQAOA leverages quantum-centric HPC platforms to update a global solution iteratively, demonstrating that distributed computing augments quantum optimization for practical problem sizes.

HADOF and DQAOA overcome standard QAOA scalability limits via decomposition. DQAOA relies on explicit partitioning and parallel aggregation, excelling on HPC or distributed platforms. HADOF uses adaptive, iterative refinement with a probabilistic global view, reducing quantum requirements per step. While DQAOA is optimal for raw parallelism and wall-clock minimization, HADOF provides

efficient sequential scaling and solution diversity. Both frameworks represent the cutting edge of distributed quantum optimization, and hybrid approaches combining their strengths are promising future directions.

3. Approach

General Overview

HADOF proceeds iteratively::

1. Encode the full QUBO as a Hamiltonian.
2. Apply an optimisation algorithm (like QAOA or QA) that produces a probability distribution to sample approximate solutions.
3. Approximate sub-Hamiltonians using the marginal probability of the binary variables (qubits).
4. Solve each sub-Hamiltonian iteratively.
5. Aggregate sampled solutions from sub-Hamiltonians to guide the next iteration.

Implementation Details

HADOF introduces a new problem decomposition method, leveraging probabilistic state information, using an iterative refinement mechanism similar to classical optimisation loops.

Let the global problem be represented in the standard QUBO form:

$$\min_{x\in\{0,1\}^n} \quad x^T Q x \tag{1}$$

where $Q \in \mathbb{R}^{n\times n}$ is an upper triangular cost matrix, and n is the dimensionality of the binary decision variable x. The corresponding quantum Hamiltonian H_Q encodes the QUBO in the computational basis.

To scale the optimisation process, H_Q is decomposed into a set of sub-Hamiltonians, each defined over a subset of the full variable set. Let $S_i \subset \{x_1, ..., x_n\}$ denote the variables of subproblem i, with $\|S_i\| = k << n$. The sub-Hamiltonian H_i is defined by:

$$H_i = \mathbb{E}_{x_{\bar{S}_i}} = P(H_Q \big| x_{S_i}, x_{\bar{S}_i}) \tag{2}$$

Here, $\bar{S}_i$ denotes the complement of S_i, and $P(x_{\bar{S}_i})$ is a distribution over unsampled variables. We use $\mathbb{E}[x_k] = P(x_k)$ as the marginal probability that variable x_k is 1, estimated from previous iterations or prior knowledge. Ideally, this expectation is estimated using a weighted average over all states the rest of the QUBO can assume. In this study, $\mathbb{E}[x_k]$ is approximated as the expected value of each qubit, by sampling it.

This transformation embeds global context into each subproblem while keeping the computational cost tractable. To estimate $\mathbb{E}[x_i]$, we use a modified QAOA and SA procedure. We use the same β-schedule for both QAOA and SA.. For each subproblem i, a QAOA circuit is constructed using the cost and mixing unitaries, as in Figure 1:

$$U(H_i, \gamma) = e^{-m\gamma_m H_i} \tag{3}$$

$$U(M, \beta) = e^{-m\beta_m \sum_{j=1}^{k} X_j} \tag{4}$$

Here, we implement QAOA as a trotterisation of QA, using the Annealing Parametrisation [17], to avoid the classical optimisation loop required to find β_m and γ_m. We start in the ground state $|+\rangle^{\otimes k}$ of the mixer Hamiltonian X and move to the ground state of the cost Hamiltonian H_i slowly enough to always be close to the ground state of the Hamiltonian, as in Figure 2. This rate is determined by the number of layers p. We initialize the β_m and γ_m in this way, moving β_m from 1 to 0 and γ_m from 0 to 1.

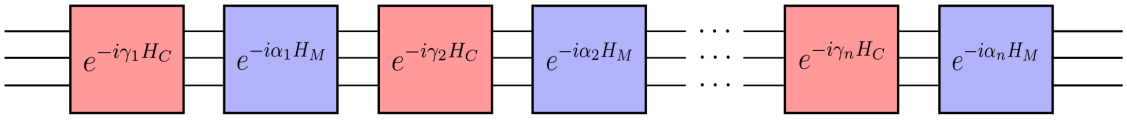

Figure 1: Standard QAOA circuit with alternating cost and mixer Hamiltonians[14]. The output produces a probability distribution over the solution space which can be samples with the shots parameter of the quantum simulation or backend. Higher sampled solutions are more likely to be solutions with better objective value.

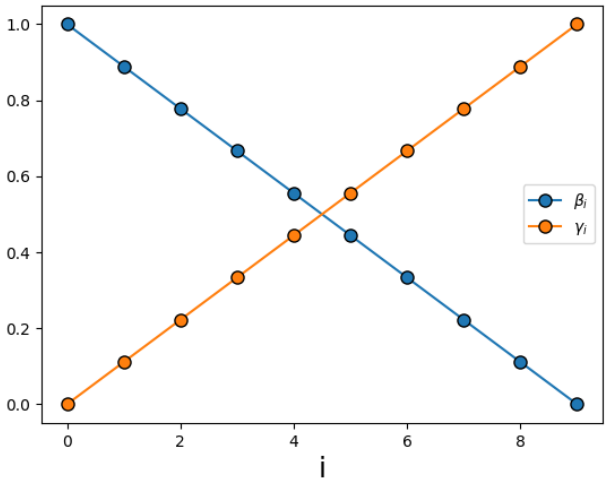

Figure 2: Trotterised QAOA parameters based on Ref [14], [17]. We move β_m from 1 to 0 and γ_m from 0 to 1 allowing the system to stay close to the ground state of the mixer Hamiltonian to the Cost Hamiltonian.

However, instead of applying the QAOA procedure completely, two changes are made to iteratively estimate the sub-Hamiltonians. In each iteration of the loop, every sub-Hamiltonian is solved using QAOA, however, the whole circuit is not applied. In the l^{th} iteration, only layers 1 to l are applied.

To approximate the value of $\mathbb{E}[x_i]$, individual qubits are sampled instead of sampling from all possible solutions of the QAOA in every iteration. The average for each qubit is used as a proxy for $\mathbb{E}[x_i]$. We follow the same procedure for beta scheduling while using SA HADOF. The optimisation process unfolds over p global iterations as in Figure 3.

4. Parametric Details

Step-by-step Procedure (Pseudocode)

1. **Initialisation:**
   - Create a global probability vector $P(x_k) \in \mathbb{R}^n$, with all values initialised to 0.5, except for the first subset S_i
2. **Iterative QAOA Loop:** For $l = 1, 2, \ldots, p$:
   a) Initialise QAOA circuit with l layers with the first l values of β_m and γ_m
   b) For each model $i \in \{0, \ldots, M-1\}$:
      - Replace inactive variables $\bar{S}_i$ by fixed expected values from previous iterations
      - Construct sub-QUBO for subset S_i using the expected values $P(x_{\bar{S}_i})$ from previous iteration.

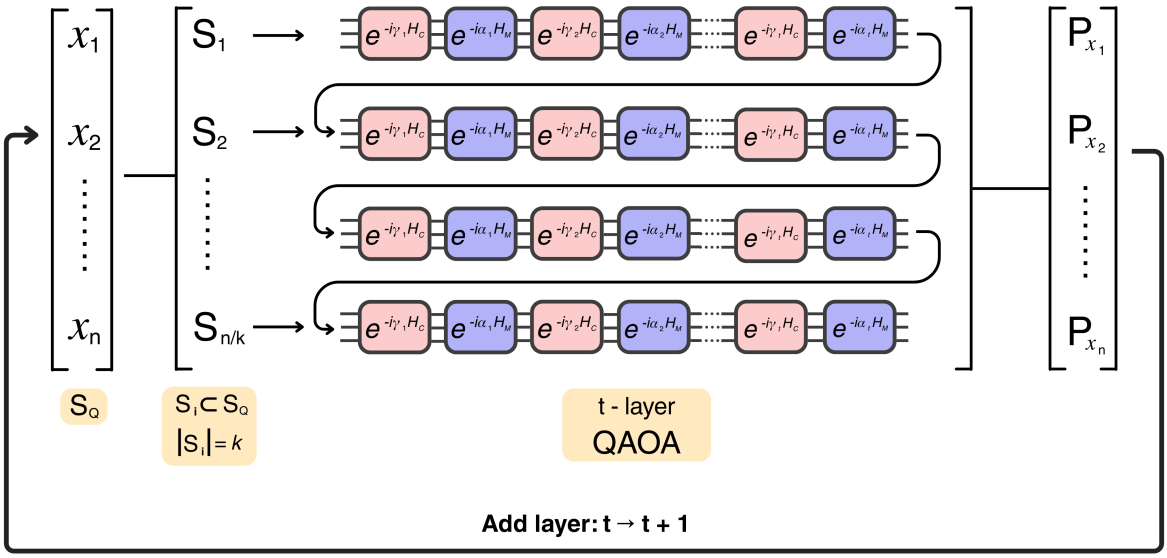

Figure 3: General overview of the HADOF framework. Here, we use QAOA as the optimiser, which is called iteratively. **(1)** We choose subsets of sizes 5 and 10 from the binary variables of the global problem. **(2)** These are used to form the sub-Hamiltonians using $P(x_i)$, approximated as the expected value of each qubit. **(3)** The QAOA circuit set up with $t = 1$ layers and in every iteration we add a layer. In this study, we use 10 layers. **(4)** Once the n/k sub-Hamiltonians are optimised, we sample them and use an aggregation policy to form the global solution probability distribution.

- Convert to Ising form: $Q \rightarrow (h, J)$ and get the sub-Hamiltonian corresponding to the sub-QUBO
- Apply QAOA circuit of depth l on k qubits where $k = \|S_i\|$:
- Update $P(x_k)$ vector by measure expected values of each qubit for current model:

$$P(x_i) = \mathbb{E}[x_i] \tag{5}$$

3. **Final Output:** After the final iteration, run full QAOA with full depth - all the layers in the beta schedule - to extract binary samples. Collect and store final solutions from all models. These solutions and their probabilities can be aggregated to form the global solution.

We generate QUBO problems by filling an upper triangular matrix using a uniform random number generator between -10 and 10. We present comparisons with CPLEX for problems with $n = 10, 20, ..., 100$ binary variables, and scale up to larger problems of size $n = 100, 200, ..., 500$ variables for the SA and HADOF methods. We choose $k = 5$ and $k = 10$, where number of QAOA and SA circuits per iteration will be n/k.

We initialise $P(x_i) = 0.5$ for all i. Circuits use 10 layers with $\beta_m = 1 - (m/10)$ and $\gamma_m = m/10$. After each sweep of the n/k circuits, we add one layer. To measure the individual qubits to update $P(x_i)$ we use 500 shots per qubit.

Finally, we sample each circuit over all k qubits using 5000 shots per circuit, to produce a distribution over each sub-solution. In this study, we only aggregate the solutions in a rudimentary manner. We form 5,000 global solutions by concatenating sampled sub-solutions in sampling order and then evaluate their objective values.

5. Results and Discussion

We evaluated HADOF on randomly generated QUBO instances of varying sizes. We compared its performance with Pennylane [2] classically simulated QAOA circuits, SimulatedAnnealingSampler

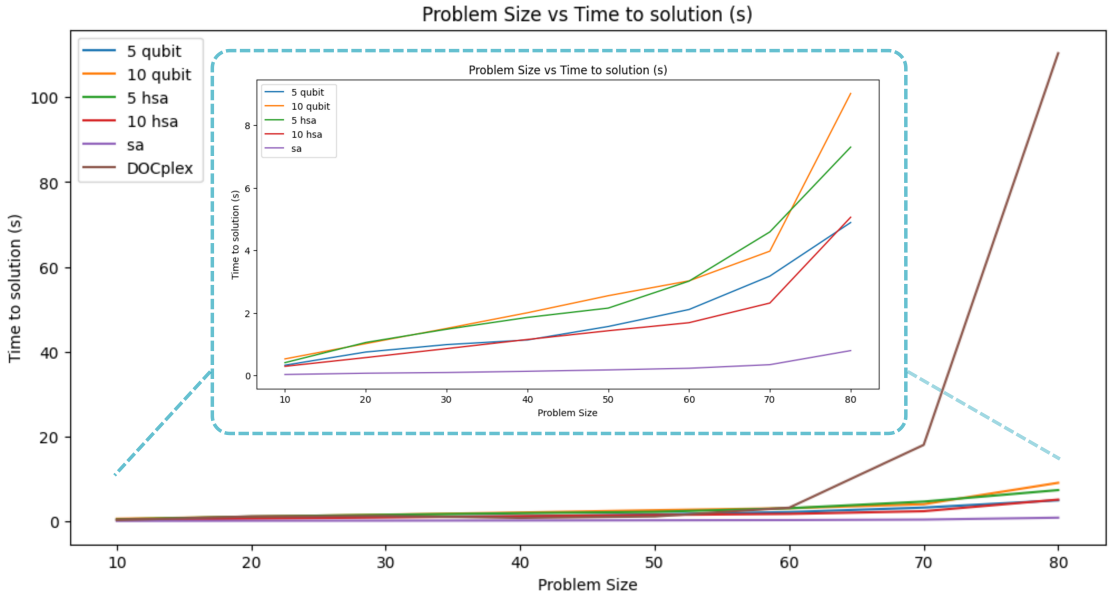

Figure 4: Time to solution as a function of problem size for CPLEX (exact classical solver), SA, HADOF QAOA and SA with 5 variable and 10 variable sub-QUBOs for problem sizes $n = 10$ to $n = 80$. CPLEX exhibits exponential scaling. SA scales the best in time as problem size increases. The inset shows the other algorithms, excluding CPLEX for clarity.

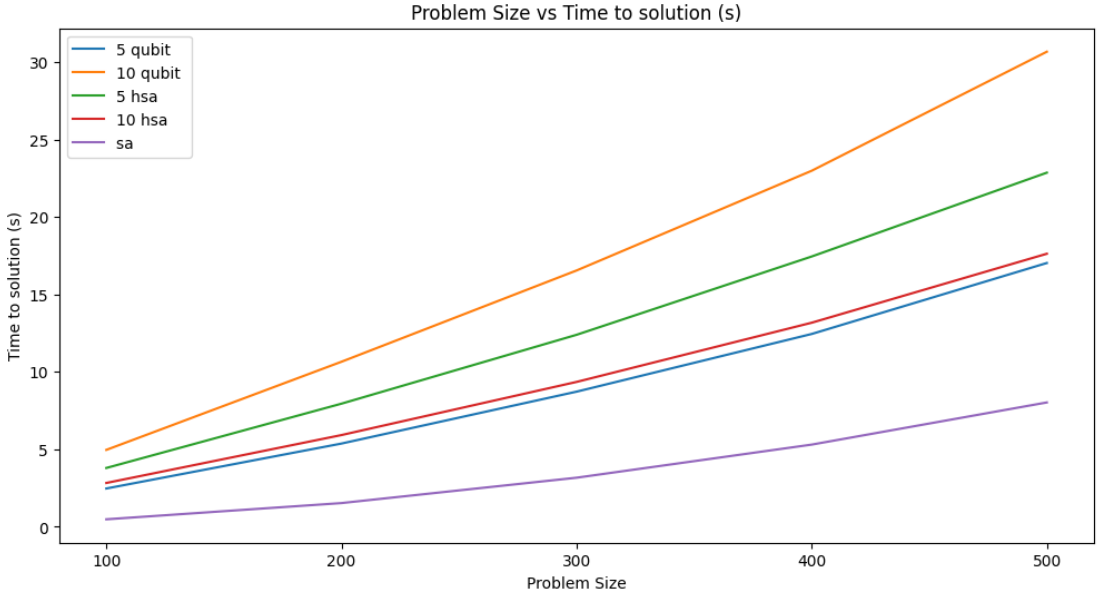

Figure 5: Time to solution as a function of problem size for SA, HADOF QAOA and SA with 5 variable and 10 variable sub-QUBOs for problem sizes $n = 100$ to $n = 500$.

from the D-Wave Ocean SDK [2] and the classical IBM Cplex solver [4].

For each problem size, we generated 100 independent QUBO instances. The 5 and 10 qubit HADOF QAOA, 5 and 10 variable HADOF SA, global problem SA and global problem CPLEX methods were run on identical problem sets. This allows us to compare SA on the global problem directly against SA using HADOF. The results were scaled such that the CPLEX objective value was set to one for problem sizes from 10-80. Beyond 80 variables, CPLEX became intractable and the solutions are scaled such that SA objective value is set to 1. We perform all the simulations on an Apple M3 Pro device. We produce a distribution of 5000 sample solutions for each algorithm, except CPLEX. This allows us to calculate

[2]D-wave ocean software, available at: https://docs.ocean.dwavesys.com/, last visited: January 7, 2026

the average solution objective value of the distribution. We also define two individual solutions from these as best objective value and most probable objective value. The best objective value is the solution with the best objective from the 5000 global solutions. The most probable solution is defined as the aggregation of the most probable sub-solutions from each circuit. These values are used to compare the accuracies of the algorithms.

Scalability and Runtime Figure 4 shows the *time to solution* for CPLEX, SA and the 5-qubit and 10-qubit HADOF approaches using QAOA and SA for 10-100 variable problems. The classical CPLEX solver demonstrates exponential scaling with problem size, as expected for exact solvers on NP-hard combinatorial problems [10]. CPLEX is nearly instantaneous up to 40 variables, but runtime rapidly increases at larger sizes. In contrast, the four HADOF approaches and SA display better scaling even upto problem sizes of 500 as shown in Figure 5. This indicates that HADOF can outperform exact solvers like CPLEX in runtime for moderate sizes, even when QAOA is classically simulated. The 5-qubit QAOA version is consistently faster than the 10-qubit QAOA. This could be because simulation of QAOA classically is expensive as the circuit size increases. We see that the 10 variable (HADOF SA) HSA is faster than the 5 variable HADOF. We note that SA takes the least time to solve all of the problems.

Solution Quality Figures 6, 7, and 8 display the *scaled objective values* for the most probable solutions, best solutions and average solutions across the algorithms respectively. These objective values are scaled to CPLEX solution for the 10-80 variable problems and scaled to SA solution for 100-500. Across all problems from 10-80, SA and CPLEX find the most optimal solution. The HADOF methods initially decrease in accuracy of best and most probable solutions as the problem size increases (10-80), but their average accuracies tend to stay stable around 0.86 and 0.90 for the 10 and 5 qubit QAOA. It stays above 0.98 using HSA. The sampling-based nature of HADOF preserves not only high solution quality but also solution diversity, as in SA, which is valuable in practical combinatorial settings [5].

Modularity HADOF is a framework that uses an optimisation process within it, to scale up the problem sizes that can be solved by it. In this research, we tested it using SA and QAOA. The framework requires that the algorithm produces a probability distribution over the solution space, from which solutions can be sampled - where good solutions are more likely to be sampled. Similar algorithms such as QA and FALQON [13] may be compatble with HADOF as well.

Testing on a Real Device We generated a single 20-variable QUBO and executed HADOF QAOA on IBM's cloud-accessible quantum device through QiskitRuntimeService [7] using 5-qubit circuits. Using the same beta scheduling parameters with 10 layers resulted in a solution with 0.84 objective value of the CPLEX solution. It took 6m and 42s to run including the classical calculation of sub-Hamiltonians and the queueing time on the real device. The Pennylane circuits were directly executable by changing the backend. Further rigourous evaluation is required to understand how HADOF performs on real NISQ devices, and with larger problem sizes.

Summary HADOF achieves hardware-efficient optimization by requiring only small quantum circuits regardless of global problem size, scaling to $n = 500$ and beyond. The framework delivers not only near-optimal objective values but also a diversity of high-quality solutions, thanks to its iterative and sampling-based design. HADOF is also modular and may be able to improve the scalability of many different algorithms.

6. Conclusion and Future Work

We introduced HADOF, a Hamiltonian Auto-Decomposition Optimization Framework, and demonstrated its capability to solve large-scale QUBO problems efficiently by iteratively dividing them into tractable subproblems. Our results show that HADOF outperforms the classical CPLEX solver in runtime

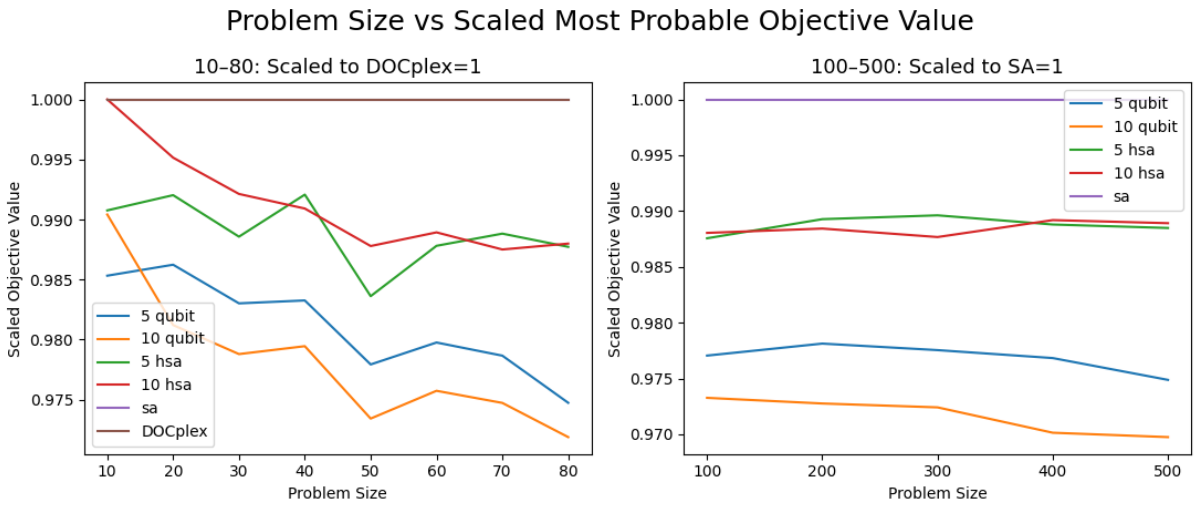

Figure 6: For small problems, we note that SA finds the exact solution offered by CPLEX all the time. The HADOF based methods decrease in scaled objective value. However for larger problem sizes, they stay around 0.98 for HSA and around 0.975 for QAOA, with the 5 qubit circuits performing better than the 10.

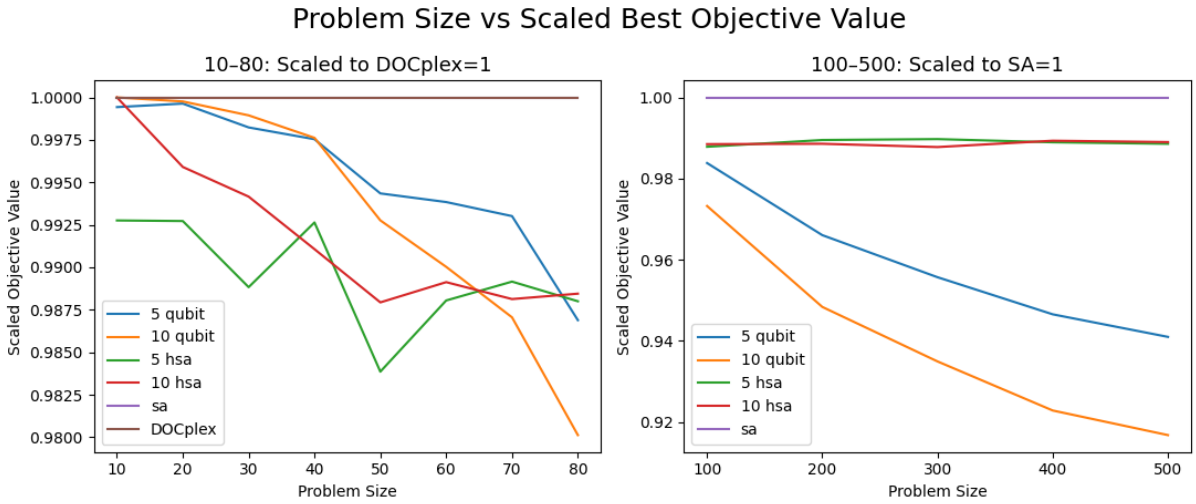

Figure 7: The HADOF based methods decrease in scaled objective value for small problem sizes. However for larger problem sizes, they stay around 0.98 for HSA but keep reducing for QAOA, with the 5 qubit circuits performing better than the 10 for all problem sizes.

and scalability, solving problems up to 500 variables that are otherwise infeasible for CPLEX under the same hardware constraints. Notably, HADOF maintains near-optimal solution quality and delivers multiple high-quality solutions in a single run.

HADOF offers several potential advances for quantum optimisation. It is extremely hardware efficient by taking advantage of only $k << n$ qubits at any time to explore a high-dimensional QUBO space, allowing NISQ based algorithms to explore large problems irrespective of qubit availability. It yields a distribution of high-quality solutions instead rather than a single optimum. Another interesting perspective of exploration could use HADOF to understand QUBO decomposition, as it iteratively creates sub-QUBOs that can be optimised separately and aggregated. This could be useful for improving or parallelising even classical algorithms which produce a distribution of solutions such as SA.

HADOF is also modular with many different optimisation algorithms that can be used under the framework to scale up beyond the available number of qubits or other device limitations that restrict the number of variables that can be solved at once.

Another key finding is that HADOF based QAOA is highly scalable even while simulating it on a

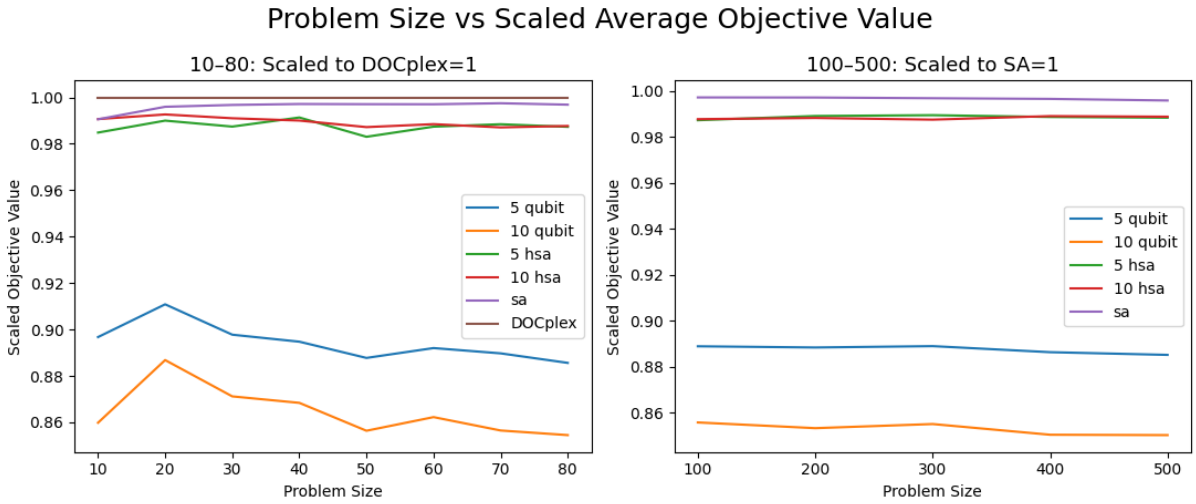

Figure 8: For small problems, we note that SA produces a distribution of 5000 solutions that is near optimal. The HADOF based methods exhibit stable average objective values with above 0.85 for all methods.

classical device. We are able to solve large size problems beyond classical solver limits (e.g., Cplex) on the same machine, in simulation. Real device implementation of HADOF may show speedups even over fast and approximate classical algorithms like SA HADOF based SA. It would be useful to study how HADOF fares against classical approximate and heuristic solvers.

While our simple aggregation—combining subproblem samples—was sufficient to surpass classical solvers in some regimes, future work will develop more robust policies to assemble sub-Hamiltonian samples into a coherent global distribution. We anticipate that adopting ideas from distributed QAOA (DQAOA) [8] and multi-level frameworks [12], such as adaptive coarse-to-fine decomposition and weighted aggregation, will allow us to better capture variable dependencies and further improve global sampling.

Following a similar sub-problem selection and aggregation strategy as in DQAOA [8] may help parallelise HADOF to run on multiple classical or quantum cores simultaneously. This ability to run large problems using small circuit sizes in embarrassingly parallel loops may allow us to further speed up and scale up the problems we can solve on current NISQ hardware.

Our results are based on simulated quantum circuits. Validation on real gate-based and annealing hardware is needed to quantify potential advantages in scalability and speed. It is important to quantify how the algorithm is affected under noisy NISQ hardware.

In addition, HADOF's decomposition scheme can be leveraged as a general divide-and-conquer technique for large QUBO problems. We plan to explore its use as a modular component within hybrid quantum-classical solvers, extending its scalability to industry-scale optimization. As quantum hardware advances, deploying HADOF with larger sub-circuits will also be investigated. Ultimately, our goal is to integrate enhanced aggregation strategies and multi-level learning to realize a fully scalable quantum-classical hybrid solver capable of addressing practical, large-scale combinatorial optimization.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Grammar and spelling check, Paraphrase and reword. After using this tool, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] Amira Abbas et al. “Challenges and opportunities in quantum optimization”. In: *Nature Reviews Physics* 6.12 (Dec. 2024), pp. 718–735.

[2] Ville Bergholm et al. *PennyLane: Automatic differentiation of hybrid quantum- classical computations*. Nov. 2018.

[3] Sergey Bravyi et al. “Obstacles to State Preparation and Variational Optimization from Symmetry Protection”. In: *Quantum* 4 (2020), p. 204.

[4] IBM ILOG Cplex. “V12. 1: User’s Manual for CPLEX”. In: *International Business Machines Corporation* 46.53 (2009), p. 157.

[5] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. *A Quantum Approximate Optimization Algorithm*. 2014.

[6] Austin Gilliam, Stefan Woerner, and Constantin Gonciulea. “Grover Adaptive Search for Constrained Polynomial Binary Optimization”. en. In: *Quantum* 5.428 (Apr. 2021), p. 428.

[7] Ali Javadi-Abhari et al. *Quantum computing with Qiskit*. 2024.

[8] Seongmin Kim et al. “Distributed Quantum Approximate Optimization Algorithm on a Quantum-Centric Supercomputing Architecture”. In: *arXiv preprint arXiv:2407.20212* (2025).

[9] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. “Optimization by Simulated Annealing”. In: *Science* 220.4598 (1983), pp. 671–680.

[10] Gary Kochenberger et al. “The unconstrained binary quadratic programming problem: a survey”. In: *J. Comb. Optim.* 28.1 (July 2014), pp. 58–81.

[11] Andrew Lucas. “Ising formulations of many NP problems”. In: *Frontiers in physics* 2 (2014), p. 5.

[12] Filip B Maciejewski et al. “A multilevel approach for solving large-scale qubo problems with noisy hybrid quantum approximate optimization”. In: *2024 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE. 2024, pp. 1–10.

[13] Alicia B. Magann et al. “Feedback-Based Quantum Optimization”. In: *Phys. Rev. Lett.* 129 (25 Dec. 2022), p. 250502.

[14] Alejandro Montanez. *Quadratic Unconstrained Binary Optimization (QUBO)*. https://pennylane.ai/qml/demos/tutorial_QUBO. Date Accessed: 2025-07-05. Feb. 2024.

[15] Humberto Munoz-Bauza and Daniel Lidar. “Scaling Advantage in Approximate Optimization with Quantum Annealing”. In: *Phys. Rev. Lett.* 134 (16 Apr. 2025), p. 160601.

[16] Christos H. Papadimitriou. “The Complexity of the Lin–Kernighan Heuristic for the Traveling Salesman Problem”. In: *SIAM Journal on Computing* 21.3 (1992), pp. 450–465.

[17] Vishal Sharma et al. *OpenQAOA – An SDK for QAOA*. 2022.

[18] Sei Suzuki. “A comparison of classical and quantum annealing dynamics”. In: *J. Phys. Conf. Ser.* 143 (Dec. 2009), p. 012002.

[19] Zeqiao Zhou et al. “QAOA-in-QAOA: Solving Large-Scale MaxCut Problems on Small Quantum Machines”. In: *Phys. Rev. Appl.* 19 (2 Feb. 2023), p. 024027.