The AI system AlphaZero\textsuperscript{1,2} famously learned to play complex and strategic games such as Go and chess and achieved super-human performance in these tasks. AlphaZero uses a combination of searching through the possible states of a game (Monte Carlo tree search) and a neural network to guide that search. This is an algorithm that learns to choose sequences of moves that lead to a victory. Chen and colleagues\textsuperscript{3} set out to employ this same set of techniques to solve a specific class of combinatorial optimization problems—the 3-satisfiability (3-SAT) problems.

3-SAT is an NP-hard problem for which the task is to decide whether there exists a choice of $n$ binary variables $b_{1…n}$ that simultaneously satisfy a set of $m$ clauses of three variables each. A clause for a 3-SAT problem is of the form $C = (b_i \text{ OR} b_j \text{ OR} b_k)$. The authors focus on the case of $m/n = 3$, for which there are always three times as many clauses as there are variables. This ratio is smaller than the critical value of $m/n = 4.2$, above which the ratio of satisfiable expressions drops to zero\textsuperscript{4}, however, it corresponds to a set of hard instances characterized by a unique solution. The authors provide a dataset that has one or several of such instances for different sizes of 3-SAT.

A 3-SAT problem of $n$ variables can be encoded into a Hamiltonian of $n$ qubits, spanning a Hilbert space of dimension $N = 2^n$, whose ground state provides the solution. Solving the 3-SAT problem is then equivalent to finding that ground state, for which several algorithms exist. The algorithm of choice here is the technique of quantum annealing\textsuperscript{5}.

In quantum annealing, the ground state is found by starting in a known (and easily prepared) ground state of an initial Hamiltonian $H_0$, and then slowly (adiabatically) interpolating to the desired final Hamiltonian $H_f$. That is, we perform $H(t) = (1 - s(t))H_0 + s(t)H_f$, for time $t$ going from 0 to $T$, where $T$ is the total annealing time, and $s(t)$ satisfying $s(0) = 0$ and $s(T) = 1$. Finding the optimal annealing path—the actual form for $s(t)$—is the central task. Starting from the ground state of $H_0$, and changing $s(t)$ adiabatically will keep the system in the instantaneous ground state of the full Hamiltonian $H(t)$. At $t = T$ then, we will have obtained the ground state of $H_f$ and hence the solution to our 3-SAT problem. At the same time, going slowly means the annealing takes more time, and we thus have an inherent trade-off between speed and accuracy: the perfect place for an optimization algorithm to help out. Following ref. 3, we choose the fidelity as the figure of merit for the accuracy of the algorithm, which simply measures the overlap of the solution obtained with the true solution (which, for benchmarking purposes, is known). A fidelity of 1 means that the algorithm achieved the perfect solution, whereas a fidelity of 0 is completely off. For more general purposes, when the ground state is not known and it might be degenerate or quasi-degenerate, the energy is a better figure of merit. For the specific problem at hand, however, we checked that the fidelity and the energy provide equivalent estimates of the algorithm’s accuracy.

The way Chen and colleagues approach the problem of optimizing $s(t)$ is by expanding it as a Fourier series with $M$ frequencies $\omega_k = \pi k / T$, and then optimizing the choice of these $M$ Fourier coefficients:

$$s(t) = \frac{t}{T} + \sum_{k=1}^{M} x_k \sin(\omega_k t).$$

The combination of the fixed linear term $t/T$ and the sinusoidal functions ensures that $s(0) = 0$ and $s(T) = 1$. The real-valued coefficients $x_k$ were not taken to be continuous values, but they are rather discretized into 40 steps between $-0.2$ and $+0.2$. Finding the right value for each $x_k$ is then a search problem similar to a board game, where the game consists of just $M$ moves, and each move means picking one of the 40 possible values. This formulation sets the problem up for a solution with an algorithm analogous to AlphaZero. We keep this structure only for

\textsuperscript{1}Niels Bohr Institute, University of Copenhagen, Copenhagen, Denmark. \textsuperscript{2}Lorentz Institute and Leiden Institute of Advanced Computer Science, Leiden University, Leiden, the Netherlands. \textsuperscript{3}e-mail: evert.vn@lorentz.leidenuniv.nl
the Monte Carlo tree search (MCTS) optimization, whereas we consider $x_i$ to be continuous and unbounded for gradient-descent methods.

### Comparing annealing methods

In Fig. 1 we compare different methods for the optimization of $s(t)$: a linear schedule for $s(t)$, a gradient-based optimizer and MCTS. The error bars are obtained as the variance of the optimization over all 18 provided instances of $n = 11$. As the original code base does not allow for easy reusability of the full MCTS + neural network algorithm, we leave it out of the comparisons for the rest of this work.

Compared with the original figure (see Fig. 3a in ref. 3), we replace the authors’ gradient-based optimization routine with a more standard implementation. For that, we choose the scipy.optimize version of the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm. For the sake of reusability, we also make several more changes to the codebase provided in ref. 3. In particular:

- We turn the hard-coded number of frequency components $M = 5$ into a parameter.
- The convergence criterion for MCTS was set to a fidelity of 0.7, which we change to either 0.99 or to whenever the optimization does not produce a change in the fidelity of more than 1% over 20 steps.

We will not include the neural network addition to the MCTS for the rest of this report. In its current form, the code base does not easily allow such re-use of the full QuantumZero algorithm.

The gradient-based algorithm shown in Fig. 1 performs much better than the original stochastic descent. The author’s version of stochastic descent evaluates gradients and updates $x_i$ sequentially, moving in orthogonal directions in the loss-landscape. Broyden–Fletcher–Goldfarb–Shanno evaluates gradients for all $x_i$ at once in the direction that minimizes the overall cost function. Although the BFGS algorithm on average leads to better fidelities than MCTS, we observed that it is more prone to remain trapped in low-quality local minima when the annealing time is large. On individual 3-SAT instances, MCTS may converge to a better result than BFGS; we observed this for 2 out of 18 instances of the $n = 11$ dataset, corresponding to the cases in which the linear annealing performs the worst. Moreover, there is a second merit to using MCTS, which becomes more apparent when we consider metrics such as the number of function evaluations and the achieved fidelity for different numbers of frequency components.

### Frequency component dependence

We now investigate the dependence of the achieved final fidelity on the number of frequency components $M$ used in the expansion of $s(t)$ (see equation (1)). We show this result for both the BFGS and MCTS algorithms in Fig. 2a,b. The BFGS optimization method’s dependence on $M$ is only very minimal, whereas for MCTS having more frequency components leads to lower accuracy for a given annealing time $T$. Importantly, however, the overall fidelity for MCTS seems to keep increasing for increasing $T$, whereas BFGS tends to get stuck in local minima that prevent it from finding the highest fidelity (compare with the $M = 3$ data for BFGS). A better minimum can be found by adding some small noise to the frequency component of the initial guess and repeating the local optimization to facilitate the exploration of the cost function landscape. In these results, the MCTS runs ran until convergence was achieved (see above for the code modifications that highlight the convergence criteria). To improve its performance at large $M$, one can choose a more stringent convergence condition, at the likely cost of running the algorithm for a longer time.

### Comparing evaluation numbers

We next investigate the required number of evaluations of the annealing schedule. This number is a more fair comparison for practical implementations as it dictates how often an annealing experiment would have to be run. Figure 2c,d shows this metric for BFGS and MCTS again for 3-SAT problems with size $n = 11$. In the former, the number of function evaluations per repetition increases monotonically with both the annealing time and the number of frequency components. For large $T$ it seems that $N_{\text{req}}$ is reaching a plateau, suggesting that the optimization process has reached a glassy phase in which the fidelity landscape has a large number of local maxima with small performance differences, with some similarity to the result presented in ref. 9. The increase of $N_{\text{req}}$ with the number of frequency components is instead due both to the increasing dimensionality of the parameter space and the computational cost of evaluating numerically the gradient of the fidelity.

However, MCTS displays remarkable independence in the number of queries to the annealer required for convergence from both $T$ and $M$, with the exception of the data for $M = 3$. This can be linked to an inherent property of the discretized energy landscape, which might smooth out some of the fine structures present in the continuous space used for BFGS, as well as a better stability of MCTS for large search spaces. Overall, with the chosen convergence condition, MCTS still require more function evaluations than a single BFGS local minimum search, even though it might reach an advantage over gradient descent for larger system sizes, in which large annealing time and schedule optimization are fundamental for reaching good accuracy.

### MAX CUT

To study the flexibility and universality of the method, we extend the performance analysis to another common classical optimization problem, namely MAX CUT\textsuperscript{10,11} on an unweighted 3-regular graph. Given a regular graph $G = (V, E)$, where $V = \{1, 2, ..., N\}$ is the set of vertices and $E = \{(i, j)\}$ is the set of edges, the MAX CUT problem Hamiltonian reads

$$H_I = \sum_{\langle i, j \rangle \in E} \left( 1 + \sigma_i^x \sigma_j^x \right),$$

(2)

which corresponds to an antiferromagnetic Ising model on the graph $G$. As $H_I$ is diagonal in the computational basis, the initial (driving) Hamiltonian can be chosen to be $H_0 = \sum \sigma_i^z$ as in the case of the 3-SAT problem investigated in the original paper\textsuperscript{1} and in the above sections of this report.
In Fig. 3 we report the fidelity as a function of the annealing time obtained with a single gradient-based optimization of the schedule \( s(t) \), a single run of MCTS until convergence, and a linear schedule \( s(t) = t/T \). The number of frequency components \( M \). The schedule optimization is repeated \( N_{\text{rep}} \) times to avoid local minima in the parameter space. BFGS (c) and MCTS (d) have markedly different behaviours: the former requires an increasing number of queries to the quantum annealer as the annealing time and the cardinality of the parameter space \( M \) increases; MCTS instead has a resource requirement that seems independent from \( T \) and \( M \), beside the data for \( M = 3 \) that reaches convergence much faster.

**Conclusion and discussion**

In this report we compared two strategies to optimize the annealing schedule on 3-SAT instances, following ref. \(^{3}\). We considered a ratio between the number of clauses and the number of variables \( m/n = 3 \), and focused on hard instances characterized by a unique solution to the combinatorial problem. We found that a gradient-based optimization in a continuous variable space leads to better fidelity than MCTS, which, however, exhibits improvement on the original simple linear schedule. As observed for the 3-SAT problem, MCTS also requires a larger number of function evaluations to reach convergence: on average, the data reported in Fig. 3 required \( N_{\text{rep}} = 200 \) for BFGS and \( N_{\text{rep}} = 1,000 \) for MCTS, although this difference might partially depend on the details of the MCTS algorithm implementation.

In Fig. 3 we report the fidelity as a function of the annealing time obtained with a single gradient-based optimization of the schedule \( s(t) \), a single run of MCTS until convergence, and a linear schedule \( s(t) = t/T \).
with complex cost-function landscapes, in which gradient-descent optimization tends to get trapped in local extremes.

**Future directions**

Our data suggest that MCTS could become systematically better than gradient-descent optimization when a large $T$ and $M$ are needed. Typically, this would be the case when the system size $n$ is also large and adiabatic evolution is hindered by vanishing energy gaps. Hence, a careful scaling analysis of the performance with $N$ can lead to a better understanding of the possible advantages of MCTS as an annealing schedule optimizer.

Furthermore, an overall improvement could be gained by a different decomposition of the annealing schedule. Our data in Fig. 2 indicate that increasing $M$ leads to very little performance gain; a different basis set might lead to a clearer advantage of MCTS in the large $M$ regime.

Finally, the apparent stability of MCTS when the optimization problem is hard (large $T$ and $M$) suggests that it might be a good candidate as the classical optimizer in variational quantum algorithms, in which gradient-based methods suffer from the appearance of so-called barren plateaus. Recent results showed that this problem can be overcome by transferring smooth optimal schedules. The neural network-aided schedule transfer implemented in the original QuantumZero algorithm might therefore be useful to tackle such issues. The current status of the original code base does not easily allow for such an investigation unfortunately, due to the missing neural network, several hard-coded parameters and sparsity of comments. For that reason we did not cover this in our work.

The modifications and additions we made to the code base can be found in a separate repository, which includes a more modular problem setup, more modular annealing methods, the BFGS method and the code for running the MAX CUT optimization problem.

**Data availability**

The data for reproducing this work are available at https://github.com/condensedAI/quantumzero (ref. 15).

**Code availability**

The code can be found at https://github.com/condensedAI/quantumzero (ref. 15).

**References**

1. Silver, D. et al. Mastering the game of Go without human knowledge. Nature 550, 354–359 (2017).
2. Silver, D. et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science 362, 1140–1144 (2018).
3. Chen, Y.-Q. et al. Optimizing quantum annealing schedules with Monte Carlo tree search enhanced with neural networks. Nat. Mach. Intell. 4, 269–278 (2022).
4. Kirkpatrick, S. & Selman, B. Critical behavior in the satisfiability of random boolean expressions. Science 264, 1297–1301 (1994).
5. Das, A. & Chakrabarti, B. K. Quantum Annealing and Related Optimization Methods Vol. 679 (Springer, 2005).
6. Morita, S. & Nishimori, H. Mathematical foundation of quantum annealing. J. Math. Phys. 49, 125210 (2008).
7. Chen, Y.-Q., Chen, Y., Lee, C.-K., Zhang, S. & Hsieh, C.-Y. QuantumZero Code Repository (GitHub, 2022); https://github.com/yutuer21/quantumzero
8. Nocedal, J. & Wright, S. Numerical Optimization (Springer, 2006).
9. Bukov, M. et al. Reinforcement learning in different phases of quantum control. Phys. Rev. X 8, 031086 (2018).
10. Halperin, E., Livnat, D. & Zwick, U. MAX CUT in cubic graphs. J. Algorithms 53, 169–185 (2004).
11. Zhou, L., Wang, S.-T., Choi, S., Pichler, H. & Lukin, M. D. Quantum approximate optimization algorithm: performance, mechanism, and implementation on near-term devices. Phys. Rev. X 10, 021067 (2020).
12. Cerezo, M. et al. Variational quantum algorithms. Nat. Rev. Phys. 3, 625–644 (2021).
13. Uvarov, A. V. & Blamonte, J. D. On barren plateaus and cost function locality in variational quantum algorithms. J. Phys. A 54, 245301 (2021).
14. Mele, A. A., Mbeng, G. B., Santoro, G. E., Collura, M. & Torta, P. Avoiding barren plateaus via transferability of smooth solutions in Hamiltonian Variational Ansatz. Preprint at https://arxiv.org/abs/2206.01982 (2022).
15. CondensedAI/Quantumzero Code Repository (GitHub, 2022); https://github.com/condensedAI/quantumzero

**Acknowledgements**

M.W. was supported by the Villum Foundation (research grant no. 25310). This project received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement no. 847523 ‘INTERACTIONS’.

**Author contributions**

MMW performed the simulations and analyzed the data. Both authors contributed to writing the code and interpreting the data and to the writing.

**Competing interests**

The authors declare no competing interests.

**Additional information**

Correspondence and requests for materials should be addressed to Evert van Nieuwenburg.

**Peer review information**

Nature Machine Intelligence thanks the anonymous reviewers for their contribution to the peer review of this work.

**Reprints and permissions information** is available at www.nature.com/reprints.

**Publisher’s note**

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© The Author(s), under exclusive licence to Springer Nature Limited 2022