The Quantum Computing Company™

Theoretical Characterization of Adiabatic Quantum Computing and Quantum Annealing

WHITEPAPER

This whitepaper presents a brief overview of the current status of what is known about adiabatic quantum computation and quantum annealing in theoretical computer science.

Quantum Terminology

Note that no standard terminology exists for some of the formal concepts described here, and that these terms have multiple meanings in the quantum computing literature.

Adiabatic Quantum Computing (AQC) refers to a general model of computation based on Hamiltonians that apply quantum operators simultaneously to large groups of qubits. This model, analogous to a Turing machine, is used in computability theory to describe what functions can be computed, and in complexity theory to characterize abstract machines and problems.

Quantum Annealing (QA) (sometimes called *stoquastic* AQC [2]) here refers to a specific instantiation of AQC, analogous to the random-access machine (RAM) commonly used in algorithm analysis. This model captures essential properties of real-world quantum processing units (QPUs) manufactured by D-Wave.

QA implements a *transverse-field Ising model* (IM) algorithm in hardware. The algorithm $\mathcal{H}$ has four components: an initial Hamiltonian H_I operating in the x basis (the transverse field); a final Hamiltonian H_F operating in the z basis (the Ising model instance); a transition function $f(t) : 0 \rightarrow t_a$ that gradually moves $\mathcal{H}$ from H_I to H_F; and an anneal time t_a.

The final Hamiltonian H_F describes instances for the IM problem (defined on spin values $\{-1, +1\}$), which is arithmetically equivalent to the quadratic unconstrained binary optimization (QUBO) problem (defined on binaries $\{0, 1\}$). Specifically, H_F represents the objective function of instance $\mathcal{I}$ in such a way that the eigenvectors and eigenvalues of the Hamiltonian exactly match the solutions and costs of the instance. The qubits are driven by $\mathcal{H}$ to find ground and low-energy states of H_F, which correspond to optimal and near-optimal solutions to $\mathcal{I}$.

Because the known bounds on computation time for $\mathcal{H}$ are intractable to compute [2], and do not apply to noisy open-system computations, we normally consider real-world annealing QPUs to operate as heuristics rather than algorithms. Here, however, we focus on abstract algorithmic properties.

Computability Theory

Computability theory considers the fundamental limits of what can be computed by machines, or equivalently, what computations can be expressed in a given formal notation system. It is widely believed that no model of computation is more powerful than a Turing machine in terms of what can be computed.

A model that can express and solve any Turing-computable function is called *Turing complete.* If it is Turing complete and fully programmable—for example if it contains a *universal circuit*, a logic circuit that can be programmed to simulate any other logic circuit,

up to a given size limit—then it is a *universal Turing machine* (UTM). See Bun [4] for details.

Note that a standalone universal circuit is not a computer: it has no memory to preserve state, and it always halts, whereas Turing-complete models must be capable of performing infinite computations. Instead, the term *quantum computer* refers to a quantum arithmetic-logic unit (ALU) together with classical components that provide control, memory, and input/output functionality.

Both AQC and QA are UTMs. The proof for QA is straightforward: it suffices to show that the quantum ALU can compute any classical logic function up to a fixed size. Many sources show how to formulate logic circuits L as a QUBO inputs Q, in such a way that an optimal solution to Q is a correct output to L. See, e.g., Pakin [7] for details.

Many people misuse the term "universal quantum computer" to distinguish between gate-model quantum computing (GM) and QA. However, the two models—having different quantum architectures—are equivalent according to the standard definition of a universal computer.

Complexity Theory

Computational complexity theory classifies problems according to *hardness*, and machine models according to *power*, based on the computational resources (time and space) needed by a given model to solve a given problem.

A primary proof technique in this field is to show that one model of computation A can simulate another model B, using only polynomially-more resources: if so, A is at least as powerful as B. If each model can simulate the other with at most polynomial overhead, the two are said to be *polynomially equivalent*. Many open questions revolve around the difficulty of showing that A is more powerful than B, which requires a proof that it is *not possible* for B to efficiently simulate A.

In this context, the following properties have been proven:

- Aharanov et al. [1] show that AQC is polynomially equivalent to GM. Both are at least as powerful as a UTM , but whether a UTM can efficiently simulate GM and AQC has not been proven. Thus, the question of polynomial equivalence between quantum and classical models is open.
- AQC has not been proven more powerful than QA (stoquastic AQC). Biamonte and Love [3] describe modifications to $\mathcal{H}$ that are sufficient—but not proven necessary—to demonstrate polynomial equivalence. Thus the question remains open.
- *Quantum speedup* is a proof that a quantum algorithm can solve a specific problem in a way that cannot be simulated efficiently by any classical algorithm. Quantum speedup for stoquastic and nonstoquastic AQC has been demonstrated using the glued trees problem; see Gilyén et al. [5].

  Quantum speedup has a very different meaning in empirical research. Note that finite experiments (including recent demonstrations of empirical quantum supremacy using a real-world annealing QPU [6]) cannot be used to settle open questions from complexity theory.

Bibliography

[1] D. Aharanov et al., Adiabatic quantum computation is equivalent to standard quantum computation, *SIAM Journal on Computing*, Vol. 37, Iss. 1 (2007)

[2] T. Albash and D. A. Lidar, Adiabatic quantum computation, *Reviews of Modern Physics*, Vol. 90, Jan-Mar (2018)

[3] J. D. Biamonte and P.J. Love, Realizable Hamiltonians for universal adiabatic computers, *Phys. Rev. A* 78, 012352 (2007)

[4] M. Bun, Lecture Notes 12: Circuits, non-uniform computation, *CAS CS 525: Complexity Theory*, Boston University, Fall 2023. Accessed 4/2025: *//cs-people.bu.edu/mbun/courses/535_F23/lectures/ lec12.pdf*

[5] A. Gilyén et al., (Sub) exponential advantage of adiabatic quantum computation with no sign problem, *Proc. 53rd Annual ACM STOC*, June (2021)

[6] A. King et al., Beyond-classical computation in quantum simulation, *Science* Vol. 388, N. 6743, March (2025)

[7] S. Pakin, Targeting classical code to a quantum annealer, *ASPLOS'19* April (2019). Accessed 4/2025: *https://dl.acm.org/doi/pdf/10.1145/ 3297858.3304071*