Classical versus quantum algorithmic complexity

Specify the problem, the input size and the resource before claiming a speedup.

10 min read Dr. Charith Jayasekara

By the end, you'll be able to

- Separate query complexity from gate cost and wall-clock time.
- State the assumptions behind Grover's quadratic query advantage.

Count the right resource

An algorithm's complexity describes how a chosen resource grows with input size. The resource might be oracle queries, gates, memory, measurements or physical runtime. These are related, but they are not interchangeable.

For a list of N possibilities with one marked item, unstructured classical search needs order N queries in the usual black-box model. Grover search uses order $\sqrt{N}$ oracle queries for bounded success probability. This is a quadratic query improvement, not an exponential one.

Input size can change the wording

If $N = 2^n$, classical exhaustive search takes order 2^n queries and Grover takes order $2^{n/2}$. Both expressions are exponential in the bit-length n; the square-root relation in N remains the useful comparison.

Costs outside a query

Preparing inputs, constructing the oracle, compiling gates, correcting errors, running shots and extracting useful classical outputs can all matter. A formal oracle advantage does not automatically survive those costs in an application.

Small browser timing comparisons would mainly measure a classical simulator and JavaScript implementation choices. This site therefore does not present their runtime as hardware speedup evidence.

Practise

In the Grover explorer, compare one marked item among four and among eight possibilities. Count iterations and success probability. Explain what the small example verifies, then list the costs it excludes.

For a stronger claim, specify a matched classical baseline, the same task and success criterion, comparable budgets, and the uncertainty of any measured results.

Reference: IBM Quantum — Fundamentals of quantum algorithms.