Classical versus quantum algorithmic complexity

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

16 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

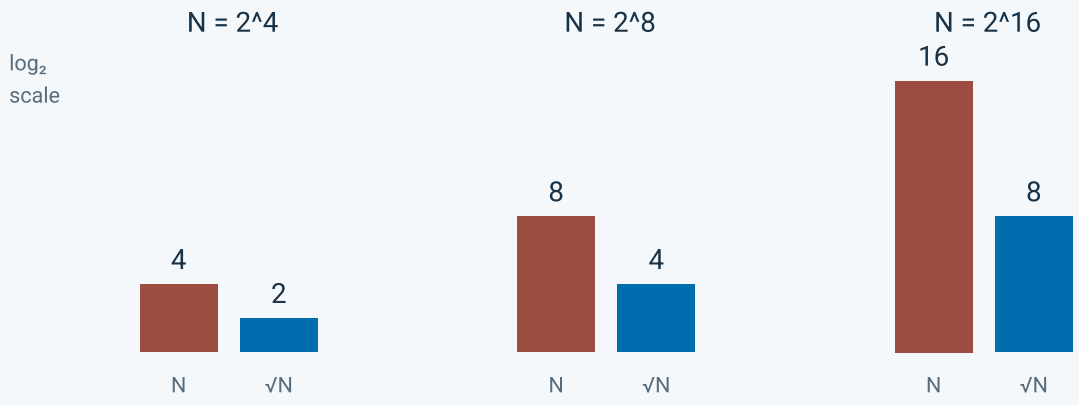
In the Grover explorer, compare one marked item among four and among eight possibilities. Count iterations and success probability. Explain what the small example verifies, then list the costs it excludes.

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

Choose the variable before reading the exponent

WORKED DIAGRAM · F23

Compare query-growth scales in the same input variable



Illustrative asymptotic scales N and √N, shown as $\log_2$(query scale). Constants, success-probability choices, oracle synthesis and physical runtime are not included. These are not measured device speeds.

The diagram plots the base-two logarithm of two query-growth expressions. Its heights represent exponents, not measured browser times. When the number of candidates doubles in bit-length, exhaustive search and square-root search both grow rapidly, but at different rates. Stating the input variable prevents an exponential-looking expression from being mistaken for an exponential advantage.

If a search space contains N candidates and its labels have n bits, then N is two to the power n. A quadratic advantage in N changes exhaustive query growth from N to its square root. Written in n, those become two to the power n and two to the power half n. Neither expression is polynomial in n. The same comparison can be described correctly in either variable if the definition is clear.

Worked example: one million candidates

Consider a black-box predicate marking one item among roughly one million. A classical unstructured search needs on the order of one million checks in the worst case, with a comparable average scale for a uniformly hidden item. Grover's query scale is on the order of one thousand. The asymptotic notation suppresses constants and the target success probability; it is not an exact instruction count.

Suppose one coherent oracle call requires a large reversible computation while a classical check is cheap. The query-count ratio alone does not determine runtime. Conversely, an expensive classical predicate may admit a useful reversible implementation. To evaluate an application, specify how the oracle is built, how its input is encoded, what precision is required and how results are verified.

For a tiny four-item case, one Grover iteration happens to succeed exactly. That special identity is pedagogically useful but does not determine performance at arbitrary N. The useful iteration count grows approximately with the square root of the search size when there is one marked item. If there are multiple marked items, their fraction changes the rotation angle and the iteration schedule.

Worst case, average case and promises

Complexity statements refer to a family of inputs and a success criterion. A worst-case guarantee can differ from typical performance on a data distribution. A promise restricts which inputs are allowed. Deutsch–Jozsa, for example, assumes a function is constant or balanced. Its conclusion does not automatically classify arbitrary functions that violate the promise.

Randomised classical algorithms also belong in the comparison. A quantum algorithm can outperform deterministic exact classical querying in a toy task while offering a less dramatic difference against bounded-error randomised querying. Always match the allowed error probability and access model. Otherwise the comparison gives one side a harder task than the other.

Separate query, gate and physical complexity

Query complexity counts access to an abstract input operation. Gate complexity includes the circuit surrounding those accesses and, when specified, the implementation of the access itself. Physical resource estimates add compilation, error correction, scheduling and control. Wall-clock performance also includes classical preprocessing, queueing and readout. These layers answer different questions.

Output requirements are another constraint. An algorithm that prepares a quantum state proportional to a solution vector has not printed every entry of that vector. Reading a large classical output can dominate the cost. Later linear-systems and machine-learning chapters therefore state whether the goal is a state, a sample or a small observable estimate. A speedup claim must match what the user actually needs to receive.

Explore scaling without timing the simulator

Use two input qubits, one marked item and zero Grover iterations. Predict its initial success probability. Reveal, then try one iteration and predict the exact four-item result. Change to three input qubits and compare successive iteration counts using the analytic rotation formula from F25. Record the probabilities, not the browser's update time.

L06 · INTERACTIVE LAB

Runs on your device

Grover search explorer

A phase oracle marks one item; diffusion reflects amplitudes about their mean. Amplification can overshoot. Oracle construction and hardware costs are excluded.

MODEL · Ideal pure-state vector, complex double precision

ORDER · q0 is the leftmost bit

Initial setup · reference state

Exact probability



Marked item probability **100.0%**

Statevector before readout · ideal simulation

Basis	Amplitude a + bi	Magnitude	Phase (rad)	Probability
|00⟩	`0 + 0i`	0.000000	undefined	0.0%
|01⟩	`0 + 0i`	0.000000	undefined	0.0%
|10⟩	`0 + 0i`	0.000000	undefined	0.0%
|11⟩	`1 + 0i`	1.000000	0.000000	100.0%

Signs and complex phases are part of the state. Phase is undefined for a zero amplitude. A shared global phase changes no probabilities; relative phases can affect a later operation. Values are rounded to six decimal places.

Prediction to try: Predict the marked item's success probability at this register size and iteration count. Explain an expected increase or decrease.

Challenge: For four items, trace all amplitudes through one iteration. For eight, compare iterations 0–6 and explain the oscillation.

Open the live experiment studio

The browser executes a classical simulation whose storage and implementation overhead differ from a quantum processor. Timing this small tool would mainly compare JavaScript paths. It can validate the algorithmic state evolution for small examples, but it cannot establish a hardware speedup. A benchmark needs an explicitly defined workload and credible classical baseline.

Practice: repair a speedup claim

A report says, "Grover reduces an n-bit search from exponential to polynomial time because it needs the square root of the candidates." Rewrite the claim with the correct variable, resource and limitation.

Check your reasoning

For one marked item among N candidates in the black-box model, Grover uses order square-root-N oracle queries for bounded success probability, compared with order N classical queries. If N equals two to the power n, the quantum query count is still exponential in n. Oracle construction and physical execution costs must be included before claiming an application runtime advantage.

Reference: IBM Quantum — Fundamentals of quantum algorithms.

Check your understanding

PAUSE & THINK

If $N = 2^n$, what is the Grover query-growth scale for one marked item?

- ☐ Polynomial in n
- ☐ Order $2^{n/2}$, still exponential in n
- ☐ One query for every N

Answer: Order $2^{n/2}$, still exponential in n. The improvement is quadratic in the number N of candidates. Query count is also distinct from total execution cost.