Noise and Resource Estimation

Build conditional resource estimates with explicit reliability and precision budgets.

Undergraduate | 22 min lesson | Dr. Charith Jayasekara

By the end, you'll be able to

- Allocate a total failure target across logical locations.
- Distinguish logical counts from code-specific physical resources.
- Compare architecture studies through their complete assumptions.

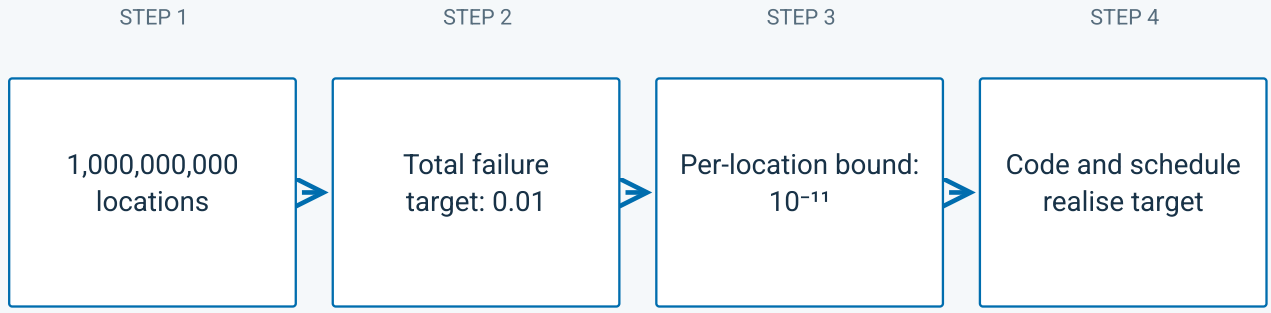
Translate an algorithm into a reliable computation

A resource estimate connects an abstract algorithm to an assumed implementation. It asks how many logical qubits, gates, measurements, code cycles and physical components are needed to meet a target success probability. A useful estimate is a conditional engineering argument with explicit assumptions. It is not a hardware delivery forecast.

Prerequisites: physical versus logical qubits, sampling and the algorithm whose cost is being estimated. This lesson develops simple arithmetic examples before discussing architecture-dependent studies.

WORKED DIAGRAM · A36

A long-circuit error budget



The union bound gives at most 0.01 failure when one billion locations each have probability at most 10^{-11}; it does not require independence.

The estimation setup

Fix the task size, output accuracy and allowed overall failure probability. Specify the logical circuit, physical error model, code family, connectivity, timing, decoder and classical reaction assumptions. Include state preparation, arithmetic,

uncomputation, routing and specialised resource-state production where relevant. Different assumptions can produce very different physical counts for the same mathematical problem.

Separate deterministic approximation error from probabilistic failure. A Hamiltonian simulation may approximate a target unitary within a norm tolerance, while syndrome failures and sampling errors have probabilistic interpretations. Combining them requires an explicit budget and a theorem or model appropriate to the task, not a casual addition of unrelated percentages.

Worked example: an error budget for a long circuit

Suppose a simplified logical computation contains one billion potentially failing locations. We want their combined failure probability at most one percent. If every location has failure probability at most one in one hundred billion, the union bound gives a total at most one percent. This bound does not require independence, though it may be loose.

Under an independent identical-failure model, the no-failure probability is one minus the per-location probability raised to the number of locations. For small products, the union-bound scale is a useful approximation. The example shows why an apparently excellent per-operation reliability can still be insufficient for a very long algorithm.

Now allocate only half the total one-percent budget to those locations, reserving the rest for state preparation and other failure mechanisms. The allowed average per-location scale halves. A physical architecture must translate that logical target into a code choice and operating schedule. A raw count of physical qubits does not perform that translation.

The union bound turns a per-location error target into a conservative total failure limit.

$$P_{\text{fail}} \le \sum_{j=1}^{L} p_j \le L p_{\text{loc}}, \qquad L = 10^9, \quad p_{\text{loc}} = 10^{-11} \quad \Longrightarrow \quad P_{\text{fail}} \le 0.01.$$

L counts potentially failing logical locations, p_j bounds location j's failure probability and p_{loc} is a common upper bound. The inequality needs no independence assumption; other failure mechanisms require their own budget.

Precision also consumes resources

Suppose one thousand approximate rotations contribute at most their individual operator errors to a conservative total bound. To keep their sum below one thousandth, allocate at most one millionth to each rotation. A synthesis method must provide that precision using the allowed discrete gate set. The logical gate count can therefore change when the output accuracy changes.

Uniform allocation is simple but not always optimal. Some rotations or subroutines are more sensitive than others, and shared error analysis may give tighter bounds. A good estimate documents the allocation and tests whether changing it materially alters the result. Precision should appear as a visible parameter rather than an unexplained constant hidden in a circuit box.

Code and architecture choices matter

An error-correcting code protects logical information through repeated syndrome extraction and decoding. The required distance or block size depends on physical error rates, error correlations, target logical reliability and the computation's duration. Logical storage, logical gates and magic-state production can have different resource requirements.

Surface-code architectures often emphasise local interactions and repeated cycles. Other codes may reduce qubit overhead while requiring different connectivity or operations. Neither can be compared fairly using only a physical-qubit headline. A primary surface-code study supplies a concrete architecture and fault-tolerance discussion, rather than a universal physical-to-logical ratio. Fowler and colleagues.

Updated RSA-2048 examples

Gidney's 2025 estimate uses fewer than one million noisy qubits and less than a week under specified nearest-neighbour, error-rate and timing assumptions. Gidney. A 2026 Pinnacle architecture study estimates fewer than one hundred thousand physical qubits using quantum LDPC codes under its own assumptions. Webster and colleagues.

These are resource analyses, not completed RSA-2048 factoring demonstrations. Their architectures and trade-offs differ, so the numbers should not be interpreted as successive measurements of one fixed machine requirement. Compare runtime, connectivity, control, decoding, resource-state production and success targets before judging which design is preferable for a particular hardware setting.

Build an estimate step by step

Start with a validated logical algorithm and small correctness checks. Count logical workspace and expensive operations, including ancillas and cleanup. Allocate approximation and failure budgets. Choose a code and physical model, translate logical operations into space and time, then schedule resource production and consumption. Include classical control latency and the observable-estimation repetitions required by the application.

Perform sensitivity analysis rather than reporting only one point. Change physical error rate, cycle time, target precision and factory throughput within plausible ranges. Identify which assumption drives the result. A table of conditional scenarios is often more useful than a single number presented with excessive precision.

Include the repetitions needed for the requested observable

A reliable coherent circuit is not always a complete algorithmic run. If the application needs an expectation to a small statistical error, it may require many independently prepared outputs. A resource estimate should multiply the per-run cost by a justified repetition count and include any postselection or amplification success factor.

This can change which subroutine dominates. Improving a unitary's gate count by ten percent may matter less than reducing a measurement budget by a factor of ten, while a very deep coherent estimation routine may trade shots for harder reliability requirements. Compare those alternatives at the same final accuracy rather than treating circuit count alone as a complete measure.

What students should report

For a small study, state the input instance, logical qubits, gate convention, approximation tolerance, assumed error model and which physical costs are excluded. If the estimate is only logical, label it as logical. If it uses a toy model, label the model rather than implying it predicts a particular device. Clear scope is more useful than a seemingly impressive hardware total.

The same habits apply outside cryptography. Chemistry, optimisation and linear-algebra estimates all depend on preparation, precision and output requirements. A favourable asymptotic algorithm can still have a large practical resource cost, while a modest algorithmic improvement can matter greatly if it reduces an expensive coherent subroutine.

Exercises and worked answers

Exercise 1. A circuit has one million locations, each with failure probability at most one billionth. What union-bound failure limit follows?

Check the worked answer

At most one thousandth, or 0.1 percent. The bound may be conservative, but it exposes the link between circuit length and required per-location reliability.

Exercise 2. Two estimates use the same logical qubit count but different code families and runtimes. Can a fixed multiplier convert either to physical qubits?

Check the worked answer

No. Encoding, logical-gate implementations, error targets, scheduling, ancillas and factories all affect overhead. A universal multiplier would hide the very assumptions the estimate is meant to analyse.

Check your understanding

PAUSE & THINK

What failure upper bound follows from 10^6 locations each at most 10^{-9}?

- ☐ 10^{-3}
- ☐ 10^{-6}
- ☐ 10^{-9}

Answer: 10^{-3}. The union bound sums the probabilities: $10^6 \times 10^{-9} = 10^{-3}$, or 0.1%.

Sources

- Fowler and colleagues
- Gidney
- Webster and colleagues