# Bernstein–Vazirani Algorithm

Recover a hidden binary string from a promised parity oracle by converting its phase pattern into a bitstring.

Core 22 min lesson Dr. Charith Jayasekara

**By the end, you'll be able to**

- Evaluate a binary dot-product function.
- Factor the hidden-string phase pattern into plus and minus states.
- Prove cancellation of every incorrect final string.

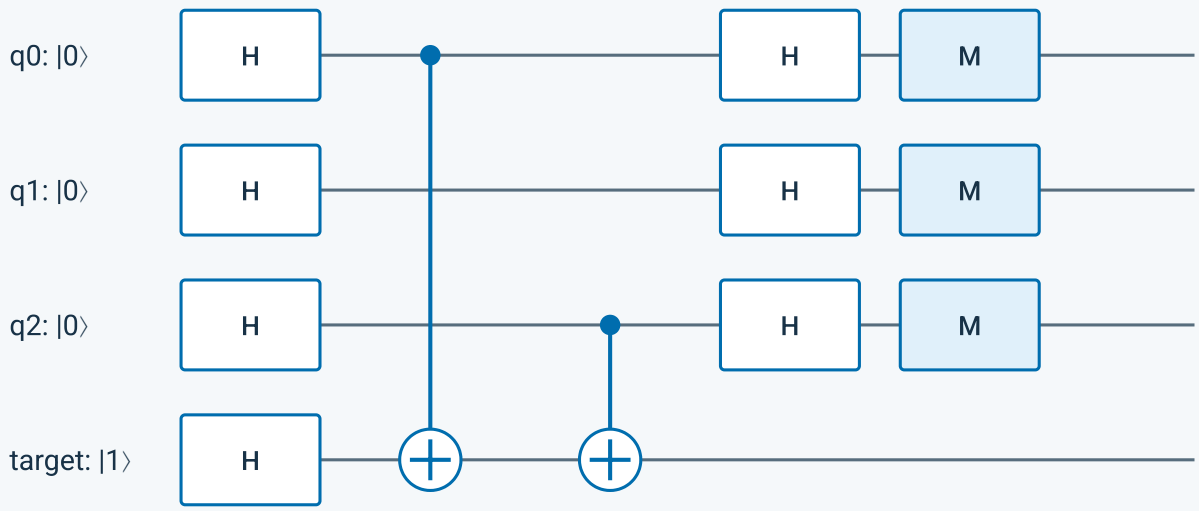
## Recover a hidden pattern, not a truth table

Bernstein–Vazirani asks you to find an unknown n-bit string s. You may query a function that returns the parity of selected input bits: multiply corresponding bits of s and x, then XOR the products. The output is one bit, even though s has n bits. A coherent quantum query and an appropriate basis change recover the entire hidden string in the ideal model.

Start with Deutsch–Jozsa and phase kickback. You need the binary dot product, which differs from an ordinary real inner product. Its arithmetic is modulo two: one plus one becomes zero when contributions are combined by XOR.

WORKED DIAGRAM · A04

### Bernstein–Vazirani: recover 101




The target starts in |1⟩ and becomes |−⟩. The CNOT gates implement this example's reversible Boolean oracle. Final input readout is 101.

## Define the promise carefully

The oracle function has the form

$$f_s(x) = s \cdot x = s_0x_0 \oplus s_1x_1 \oplus \cdots \oplus s_{n-1}x_{n-1}.$$

The goal is to return s. Input access is a reversible bit oracle or its corresponding phase oracle. A deterministic classical method can query each unit-vector input: putting one in position j and zero elsewhere returns s in that position. This uses n queries. Each ordinary query yields one bit; an exact classical strategy needs enough information to distinguish all two-to-the-power-n possible strings.

The quantum method uses one coherent query, but that oracle can contain many elementary gates. If the hidden string were explicitly supplied as an ordinary classical input, there would be no mystery to solve. The separation concerns the specified black-box access, not magical extraction of a secret from any physical system.

## Procedure and diagram

Prepare the n-qubit input uniformly with H gates. Prepare a minus target for a bit oracle, or use an explicitly supplied phase oracle. Query once, attaching phase negative one to the power s dot x. Apply H again to every input qubit. Measure the input register in Z and interpret the bitstring as s in the stated wire order.

For the three-bit example, the phase oracle factors into Z on q0 and q2, with no phase flip on q1. The diagram groups these operations as one invocation of the specified oracle interface. Its output is minus, plus, minus before the last H layer. A bit-oracle implementation instead includes the separately prepared minus target, which remains factored out. Both descriptions must preserve the same input-relative phase pattern.

The oracle writes the hidden dot product into the phase of each input label.

$$|\psi_s\rangle = \frac{1}{\sqrt{2^n}} \sum_x (-1)^{s \cdot x} |x\rangle = H^{\otimes n} |s\rangle.$$

The hidden bitstring is $s$, the queried input is $x$, and their binary dot product is defined in the promise above. The minus target is omitted because it factors out.

## Worked example: s equals 101

The function returns the XOR of the first and third input bits. Its value is zero on 000, one on 001, zero on 010, one on 011, one on 100, zero on 101, one on 110 and zero on 111. These eight values would require several ordinary queries to identify by a generic classical strategy, but their phase pattern has a compact factorisation.

After the quantum query, the input state is minus on the first wire, plus on the second and minus on the third. To see this, note that the phase factor splits into the product of a sign for the first bit and a sign for the third bit. H maps each minus to one and each plus to zero. The final state is therefore 101, with probability one.

This example is particularly useful because it does not need entanglement among the input qubits. The phase pattern factorises. Quantum advantages in a chosen oracle model cannot always be explained simply by pointing to entanglement. The access model and coherent basis transformation are central here.

Each hidden one supplies a minus state on its corresponding input wire.

$$s = 101: \qquad |\psi_s\rangle = |-\rangle \otimes |+\rangle \otimes |-\rangle \xrightarrow{H^{\otimes 3}} |101\rangle.$$

The leftmost wire is $q_0$. Applying $H$ maps minus to one and plus to zero on each wire.

### Checkpoint 1

PAUSE & THINK

**Which input state appears after kickback for s=101?**

- ☐ |+⟩|−⟩|+⟩
- ☐ |−⟩|+⟩|−⟩
- ☐ |101⟩

**Answer:** |−⟩|+⟩|−⟩. The phase (−1)^(x0+x2) factors into minus on q0, plus on q1 and minus on q2.

## Why the transform returns s

The amplitude of output z after the last H layer is the average of signs determined by the binary dot product of s XOR z with x. If z equals s, that dot product is zero for every x, so every contribution is positive and the amplitude is one. If z differs from s, choose a position where they differ. Pair each input x with the input obtained by flipping that position. The two contributions have opposite signs and cancel.

Thus all wrong outputs have zero amplitude and the correct output has unit amplitude. This pairing argument gives a clear correctness proof without expanding a large matrix. It is the orthogonality of binary Fourier characters expressed as a cancellation rule.

The final transform cancels every candidate label except the hidden string.

$$\alpha_y = \frac{1}{2^n}\sum_x (-1)^{(s\oplus y)\cdot x} = \begin{cases} 1, & y = s, \\ 0, & y \neq s. \end{cases}$$

If $y \neq s$, choose a bit where they differ and pair inputs that differ at that bit; the two signs cancel.

## Constant offsets and conventions

Suppose the oracle is affine: f(x) equals s dot x XOR a fixed bit b. The extra bit adds a common sign negative one to the power b after phase kickback. The input measurement still returns s. It does not reveal b through that readout. A separate ordinary query at zero would reveal the offset, if the task also requires it.

This is another example of output design. The phase procedure discards a global distinction while preserving the relative pattern it needs. Do not claim that recovering s means recovering every possible detail of an arbitrary function. The linear or affine promise is essential. For a nonlinear function, the final H measurement samples its Fourier spectrum rather than necessarily returning one deterministic hidden string.

An added constant changes only a common sign, so the hidden string is still recovered.

$$f(x) = s\cdot x \oplus b \quad\Longrightarrow\quad |\psi_{\text{out}}\rangle = (-1)^b|s\rangle, \qquad b \in \{0,1\}.$$

The measurement determines $s$. This procedure does not separately determine the offset $b$.

### Checkpoint 2

PAUSE & THINK

**If f(x)=s·x⊕1, how does the measured result change?**

- ☐ Every bit of s flips
- ☐ The result becomes random
- ☐ It still returns s

**Answer:** It still returns s. The output is −|s⟩. The measured hidden string is unchanged; b can be found with f(0).

## Costs and a sensible experiment

Outside the oracle, preparation and readout use a number of simple gates linear in n. The query advantage is one versus n under the given access and exact-success model. A physical implementation of a parity oracle may itself use a number of controlled gates linear in the weight of s, plus calibration and error costs. Report both the oracle count and the implementation when discussing runtime.

For a hand experiment, choose a hidden three-bit string, write the phase sign for all eight inputs, and factor the result into plus/minus states. Predict the final measurement before simulating. Test the all-zero string, a single-one string and 101. These cases catch accidental assumptions that every promised function must be balanced: the all-zero hidden string gives a constant-zero function, but the algorithm still works.

## Explore the mechanism

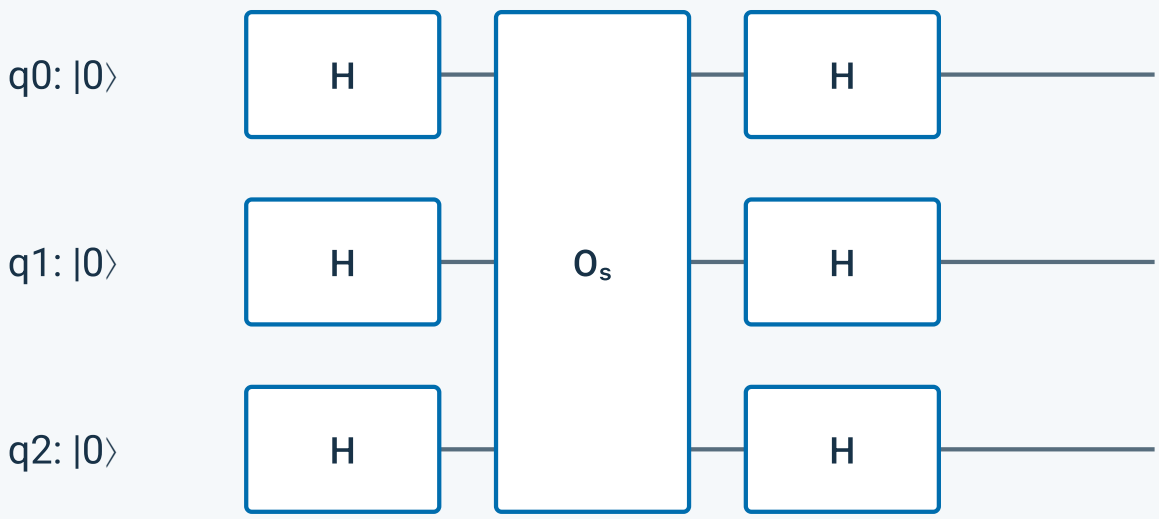
Use Explore freely to compare amplitudes and measurement probabilities. Choose Predict & test for a quick prediction, then run measurements. Change one setting at a time and explain the change.

ALGORITHM LAB

Ideal local model

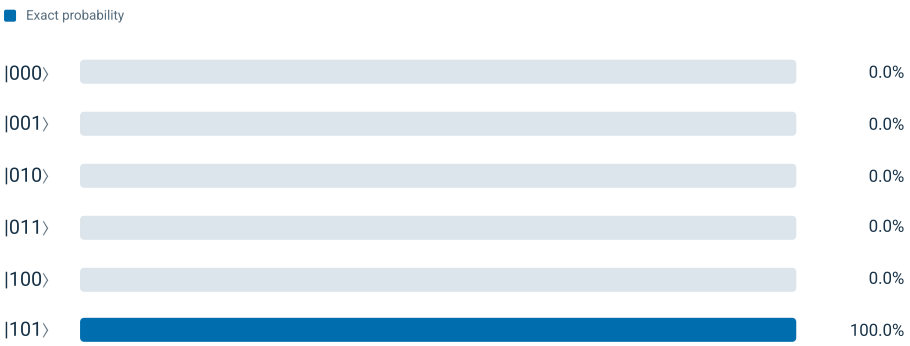
### Bernstein–Vazirani: recover a bit pattern

The phase oracle applies (−1)^(s·x), then H on each input. The chosen secret defines the simulated oracle.




The statevector is an ideal simulation. A useful application also counts state preparation, oracle construction, compilation and measurement.

### Initial model reference



|110〉 0.0%

|111〉 0.0%

**Statevector before readout · ideal simulation**

| Basis | Amplitude a + bi | Magnitude | Phase (rad) | Probability |
|---|---|---|---|---|
| \|000〉 | `0 + 0i` | 0.000000 | undefined | 0.0% |
| \|001〉 | `0 + 0i` | 0.000000 | undefined | 0.0% |
| \|010〉 | `0 + 0i` | 0.000000 | undefined | 0.0% |
| \|011〉 | `0 + 0i` | 0.000000 | undefined | 0.0% |
| \|100〉 | `0 + 0i` | 0.000000 | undefined | 0.0% |
| \|101〉 | `1 + 0i` | 1.000000 | 0.000000 | 100.0% |
| \|110〉 | `0 + 0i` | 0.000000 | undefined | 0.0% |
| \|111〉 | `0 + 0i` | 0.000000 | undefined | 0.0% |

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.

## Chapter summary

- A linear Boolean rule produces a separable pattern of plus and minus states.
- The final Hadamards return the hidden coefficient string exactly in the ideal model.
- A constant offset contributes a global phase and needs a separate query to determine.

### A common misconception

**Recovering s also recovers every part of an affine function, including its offset.** The offset is a common sign in this circuit and does not affect the measured string. One ordinary evaluation at x=0 reveals it.

## Code practice

Download the tested Qiskit notebook and follow the pinned environment instructions. Inspect the circuit, predict a result and run its independent assertions.

## Exercises

1. Let s be 011. What are f(100), f(010) and f(111)? What string should the quantum circuit return?
2. Add an affine offset b equal to one. Which final probabilities change? How could you learn b separately?

Worked answers

1. The function ignores the first bit and XORs the last two. The values are zero, one and zero respectively. The final quantum output is 011 under the book's leftmost-bit convention.
2. No input-register probabilities change because the offset contributes only a common phase. Querying the affine function at 000 returns b directly. Recovering the slope string and recovering the offset are distinct output tasks.

## Check your understanding

PAUSE & THINK

**An affine oracle adds a constant-one offset to s·x. What changes in the hidden-string measurement?**

- [ ] Every output bit flips
- [ ] Nothing; the extra sign is global
- [ ] The output becomes random

**Answer:** Nothing; the extra sign is global. The offset negates every amplitude by the same factor. The circuit still returns s; learning the offset is a separate task.

## Sources

The Bernstein–Vazirani section of IBM's query-algorithm lesson explains the linear-function promise and its use of the same Hadamard-query-Hadamard circuit.