AI explanation: \[ O\left(\sqrt{\frac{N}{M}}\right) \]

Project: Oracle in Quantum Computing  Detailed Explanation
Document: Document 1.3
Author: mujirin
Verifier: Not verified
Downloaded: June 14, 2026 05:00 KST
Status: Working

Why Grover's Oracle Count Scales Like O(√((N)/(M)))

The expression

O(√((N)/(M)))

appears in the parent document in the discussion of Grover's algorithm. It describes the number of oracle calls needed, up to constant factors, to find a marked solution with high probability when there are N possible candidates and M of them are solutions.

This is not a statement about the cost of building the oracle. It is a statement about query complexity: how many times Grover's algorithm must call the marking oracle once such an oracle is available. The parent document directly supports this interpretation by saying that Grover's algorithm alternates between a phase oracle and a diffusion operator, and that after about this many oracle calls, measuring gives a solution with high probability. The deeper reasoning behind the formula comes from amplitude amplification.

The Search Setting

Suppose there is a finite search space of size N. We can label its elements as basis states

|0⟩, |1⟩, dots, |N-1⟩

Among these N candidates, exactly M are solutions. The oracle is usually represented by a Boolean function

f(x) = begincases 1, x is a solution, ; 0, x is not a solution. endcases

The corresponding phase oracle acts as

O|x⟩ = (-1)^f(x)|x⟩

So solution states receive a minus sign, while non-solution states remain unchanged. This phase flip is not itself the answer. Its role is to create a relative phase difference that the rest of the algorithm can turn into increased probability of measuring a solution.

Grover's algorithm usually begins in the uniform superposition

|s⟩ = frac1√(N)sum_x=0^N-1|x⟩

In this state, every candidate has equal probability 1/N. Since M of the N candidates are good, the initial probability of measuring a solution is

(M)/(N).

Classically, if one samples uniformly at random and checks candidates, one expects to need on the order of

(N)/(M)

queries to find a solution. Grover's algorithm improves this to roughly the square root.

(N)/(M)).

The highlighted expression is this quadratic improvement.

Compressing the Search Space into Two Directions

The cleanest way to understand the formula is to notice that Grover's algorithm does not need to treat all N basis states separately. Because the oracle marks all solutions in the same way, and the diffusion operator treats all basis states symmetrically, the dynamics stay inside a two-dimensional subspace.

Define the normalized uniform superposition over good states:

|G = frac1(M)x:f(x)=1|x.

Define the normalized uniform superposition over bad states:

|B = frac1(N-M)x:f(x)=0|x.

assuming 0<M<N. These two states are orthogonal: |G lives entirely in the solution subspace, while |B lives entirely in the non-solution subspace.

The initial uniform state can be rewritten as a combination of these two directions:

|s = ((M)/(N))|G + ((N-M)/(N))|B.

This equation is the heart of the scaling. The coefficient of |G is

((M)/(N)).

That coefficient is the initial amplitude in the good direction. Its square,

ft(((M)/(N)))= (M)/(N),

is the initial probability of measuring a solution.

Grover's algorithm amplifies amplitude, not probability directly. This is why a square root appears.

The Angle Behind the Formula

It is common to describe the initial state using an angle , defined by

sin  = ((M)/(N)).

Then

cos  = ((N-M)/(N)),

and the initial state becomes

|s□ = sin□ |G□ + cos□ |B□

If M is much smaller than N, then □ is small. For small angles,

sin□□□.

so

□□□(√(M/N)).

A single Grover iteration consists of two reflections: the oracle reflection, which flips the phase of the good states, and the diffusion reflection, which reflects about the initial superposition. The product of two reflections is a rotation. In this two-dimensional |G□, |B□ plane, each Grover iteration rotates the state toward the good direction by an angle approximately 2□.

More precisely, after k Grover iterations, the state has the form

sin((2k+1)□)|G□ + cos((2k+1)□)|B□.

Therefore the probability of measuring a solution is

sin□((2k+1)□).

To get high probability, we want the angle (2k+1)□ to be close to □/2, because

sin□□□ft((□/(2))=1.

So we choose k such that

(2k+1)□□ (□/(2).

Solving roughly gives

k □ (□/(4□).

Since

□□□(√(M/N)),

we get

k □ (□/(4)□√(N/M)).

Ignoring the constant factor □/4, this becomes

O□ft(□√(N/M)).

That is the origin of the highlighted expression.

What the Big-O Is Hiding

The notation

$$O(\sqrt{(N/M)})$$

does not claim that the exact number of oracle calls is equal to √(N/M). It means that the number of oracle calls grows no faster than a constant multiple of √(N/M), under the usual assumptions of Grover search.

A more precise estimate for the number of Grover iterations is often written as

$$k \approx \lfloor (\pi/4)\sqrt{(N/M)} \rfloor$$

when M≪N. Each iteration uses one call to the phase oracle, plus a diffusion operation. Depending on convention, some descriptions may count additional oracle calls for state preparation, verification, or amplitude estimation, but the core Grover iteration uses one marking oracle query.

The parent document's statement is therefore best read as a query-complexity claim: assuming access to an oracle that marks the M solutions among N candidates, Grover's amplitude-amplification procedure needs on the order of √(N/M) oracle invocations to produce a solution with high probability.

Why More Solutions Mean Fewer Queries

The formula also has an intuitive interpretation. If there are more marked items, the initial uniform superposition already has more amplitude in the good subspace.

When M=1, there is only one solution, and the query complexity becomes

$$O(\sqrt{(N)}).$$

This is the familiar form of Grover's search.

When M is larger, the expression decreases:

$$O(\sqrt{(N/M)}).$$

For example, if M=N/100, then one percent of the search space consists of solutions. The number of Grover iterations is on the order of

$$\sqrt{(N/(N/100))} = \sqrt{(100)} = 10.$$

This matches the geometric picture: if the good subspace is already large, the initial state is already closer to it, so fewer rotations are needed.

At the extreme, if M=N, every candidate is a solution. Then no search is needed. The formula gives

$$\sqrt{(N/N)} = 1,$$

which is only a constant-order statement. In practice, one can measure immediately and obtain a solution. Big-O notation is coarse enough that it does not distinguish between zero iterations and a constant number of iterations.

Important Assumptions Behind the Expression

The expression rests on several assumptions that matter when verifying the statement in a research context.

First, it assumes an unstructured search problem. The oracle tells us whether a candidate is good, but it does not reveal additional exploitable structure. If the problem has algebraic or combinatorial structure, another algorithm may do better than Grover, or the oracle model may fail to capture the true cost.

Second, it assumes the algorithm starts in the uniform superposition over N candidates. More general amplitude amplification replaces M/N with the initial probability mass Z on good states. In that setting, the query count becomes

\[ O\left(\frac