Week Twelve PHY-480

Lewis

November 2025

1 Class 22

Q22.1

Show that the matrix operation $H0$ gives the uniform superposition state of a qubit, where H is the Hadamard matrix.

The Hadamard matrix for a single qubit is

$$H = \frac{1}{\sqrt{2}} \left(1 \right) 11 - 1.$$

Applying H to the ground state $0 = \left(1 \right)$
0 gives

$$H0 = \frac{1}{\sqrt{2}} \left(1 \right) 11 - 1 \left(1 \right) 0 = \frac{1}{\sqrt{2}} \left(1 \right) 1 = \frac{1}{\sqrt{2}} (0 + 1).$$

Hence, $H0$ produces the **uniform superposition state**, where the qubit has equal probability of being in 0 or 1.

2 Class 22

Q22.2

Give a concise statement of the amplitude amplification strategy used in designing quantum algorithms.

Amplitude amplification is a general quantum strategy for **increasing the probability amplitude** of target states within a quantum superposition while **suppressing all others**. It repeatedly applies a combination of:

- a **phase inversion** about the target state, and
- a **reflection** about the average amplitude,

thereby rotating the state vector toward the target in Hilbert space. Grover's algorithm is the canonical example, achieving a quadratic speedup over classical unstructured search.

Q22.3

Outline a Python algorithm for a simulation of the Grover algorithm on a classical computer using matrices of dimension 2^n, where n is the number of qubits.

1. Set the number of qubits n and compute $N = 2^n$.
2. Define the Hadamard matrix:

$$H = \frac{1}{\sqrt{2}} (1) 11 - 1, \quad H^{\otimes n} = Kroneckerproductof$$

n$copiesofH$.

2. Define the initial state $0^{\otimes n}$ and compute the uniform superposition:

$$\psi_0 = H^{\otimes n} 0^{\otimes n}.$$

3. Construct the target oracle:

$$U_{tar} = -I_N + 2E_{tar,tar},$$

where $E_{tar,tar}$ is a matrix with a single 1 at the target index.

4. Construct the diffusion operator:

$$D = H^{\otimes n}(200 - I_N)H^{\otimes n}.$$

5. Define the Grover operator:

$$G = D\,U_{tar}.$$

6. Apply G repeatedly:

$$\psi_t = G^t \psi_0.$$

7. After each iteration, record the probability of the target state:

$$P_{tar}(t) = |\langle tar|\psi_t\rangle|^2.$$

8. Plot $P_{tar}(t)$ versus t to observe amplitude amplification.

Q22.4

Explain the expected behavior of the output-state probability of the target state as a function of t, the number of times the Grover operator is applied.

Each application of the Grover operator G performs a **rotation in a two dimensional subspace** spanned by the target state *tar* and the average of all non-target states. The probability of measuring the target state increases sinusoidally with the number of iterations t:

$$P_{tar}(t) = \sin^2\left((2t+1)\theta\right),$$

where $\sin^2(\theta) = 1/N$ and $N = 2^n$ is the total number of states. The probability reaches its maximum near

$$t_{opt} = \left\lfloor \frac{\pi}{4}\sqrt{N} \right\rfloor,$$

after which additional iterations will begin to decrease the success probability again (overshooting the target).

Summary: The amplitude amplification process boosts the likelihood of finding the target state up to near certainty after $\sim \frac{\pi}{4}\sqrt{N}$ iterations.

Q23.1. The Quantum Phase Estimation (QPE) algorithm finds the phase θ of an eigenvalue $E = e^{2\pi i\theta}$ of a unitary operator U. It prepares a superposition in a control register, applies controlled powers of U to encode the phase, and then uses the inverse Quantum Fourier Transform to read out an n-bit estimate of θ. :contentReference[oaicite:0]index=0

Q23.2. The order r of $(a, N) = (11, 17)$ is the smallest positive r such that $11^r \equiv 1 \pmod{17}$. Checking powers shows $11^{16} \equiv 1 \pmod{17}$, and no smaller r works. Thus, the order is

$$r = 16.$$

Q23.3. The quantum order-finding algorithm determines the order r of two co-prime integers (a, N) by applying Quantum Phase Estimation to the unitary $U|k\rangle = |ak \bmod N\rangle$. QPE outputs a number close to s/r, and from this ratio we extract a candidate value of r. We then verify it by checking whether $a^r \equiv 1 \pmod{N}$.

Q23.4. A simple Python outline for quantum order finding:

```
def order_finding(a, N):
    # Search for smallest r where a^r mod N = 1
    for r in range(1, N):
        if pow(a, r, N) == 1:
            return r
    return None
```

This mirrors the structure of quantum order finding: identify the period of the sequence $a^k \bmod N$ and verify $a^r \equiv 1$.