INFORME PRÀCTICA ALGORISME SHOR

Aquest informe presenta la implementació d'una simulació completa de l'algorisme de Shor per a la factorització d'enters compostos. La pràctica s'ha desenvolupat en Python utilitzant NumPy, simulant el comportament d'un ordinador quàntic mitjançant tècniques clàssiques. Els resultats demostren la viabilitat de la simulació i permeten comprendre els fonaments de la computació quàntica aplicada a problemes criptogràfics.

1. Introducció i Objectius

L'algorisme de Shor (1994) representa un dels avenços més significatius en computació quàntica, permetent factoritzar enters en temps polinòmic $O(\log N)^3$. La seva rellevància és crítica en criptografia, ja que sistemes com RSA depenen de la dificultat de la factorització.

Objectius de la Pràctica

1. Comprendre els fonaments de la computació quàntica aplicats a la factorització
2. Implementar una simulació completa pas a pas de l'algorisme de Shor
3. Analitzar la relació entre la QFT i el càlcul del període d'una funció modular
4. Integrar la part clàssica per obtenir els factors de $N = p \cdot q$

2. Fonaments Teòrics

L'algorisme de Shor redueix la factorització al problema de trobar el període r de la funció **$f(x) = a^x \bmod N$**, on gcd(a, N) = 1. Un cop trobat r (amb r parell i $a^{r/2} \not\equiv -1 \pmod N$), els factors es calculen com:

- **$p = \gcd(a^{r/2} - 1, N)$**

- **$q = \gcd(a^{r/2} + 1, N)$**

L'algorisme utilitza tres propietats quàntiques fonamentals: **superposició**, **entrellaçament** i la **Transformada de Fourier Quàntica (QFT)**.

3. Implementació

3.1 Arquitectura del Sistema

La implementació s'estructura en la classe ShorAlgorithm que encapsula:

- Funcions auxiliars clàssiques (exponenciació modular eficient, selecció aleatòria de bases)
- Simulació del mòdul quàntic (5 components principals)
- Integració clàssica per coordinar tot el procés

Aquesta arquitectura orientada a objectes facilita la modularitat i la reutilització del codi (veure shor.py).

3.2 Components Quàntiques Implementades

3.2.1 Registre Quàntic Uniforme

La funció create_uniform_quantum_register() crea un registre de n qubits en superposició uniforme. Cada estat $|x\rangle$ té amplitud $1/\sqrt{(2^n)}$, garantint que la suma de probabilitats sigui 1. Aquesta inicialització simula l'aplicació de portes Hadamard a tots els qubits, utilitzant arrays de NumPy amb nombres complexos.

Assoliment objectiu 1: Simula correctament la superposició quàntica amb vectors normalitzats.

3.2.2 Exponenciació Modular i Entrellaçament

La funció quantum_modular_exponentiation() implementa l'operació quàntica $|x\rangle \rightarrow |x, a^x \bmod N\rangle$. Aquesta operació crea correlacions entre el primer registre (input x) i el segon registre (output f(x)). S'utilitza una estructura de diccionari que mapeja

cada valor de f(x) als seus corresponents estats x amb les seves amplituds, simulant així l'entrellaçament quàntic.

Assoliment objectiu 1: Implementa l'entrellaçament quàntic de forma eficient.

3.2.3 Mesura del Segon Registre

La funció measure_second_register() simula el col·lapse de la funció d'ona segons la regla de Born, on la probabilitat de mesurar un estat és proporcional al quadrat del mòdul de la seva amplitud. Després de la mesura, el registre es col·lapsa mantenint només els estats consistents amb el resultat observat, i posteriorment es renormalitza.

Assoliment objectiu 1: Reprodueix correctament el comportament probabilístic dels sistemes quàntics.

3.2.4 Transformada de Fourier Quàntica (QFT)

La funció quantum_fourier_transform() utilitza la FFT de NumPy per simular la QFT. La QFT és anàloga a la transformada discreta de Fourier i, per un registre amb estructura periòdica, concentra l'amplitud en estats que són múltiples enters de $2^n/r$, on r és el període buscat.

Assoliment objectiu 3: Estableix clarament la relació entre la QFT i l'extracció de la periodicitat de la funció modular.

3.2.5 Càlcul del Període amb Fraccions Continues

La funció continued_fraction_expansion() extreu el període r a partir de l'estat mesurat després de la QFT. Si es mesura l'estat k, llavors $k/2^n \approx j/r$ per algun enter j. Utilitzant la classe Fraction de Python, l'algorisme de fraccions continues recupera el denominador r amb alta probabilitat. Addicionalment, es realitzen verificacions per assegurar que r és vàlid (parell i que $a^r \equiv 1 \bmod N$).

Assoliment objectiu 3: Demostra com convertir informació quàntica en informació clàssica útil.

3.3 Integració Clàssica

La funció principal factorize() coordina tot el procés seguint l'esquema de l'algorisme:

1. Tractament de casos trivials (N parell, potències)
2. Bucle principal amb múltiples intents:
    - Escollir a aleatori amb gcd(a, N) = 1
    - Invocar el mòdul quàntic per trobar el període r
    - Verificar condicions (r parell i $a^{r/2} \not\equiv -1 \bmod N$)
    - Calcular factors mitjançant gcd

Assoliment objectiu 4: Integra correctament la part clàssica amb el mòdul quàntic per obtenir els factors finals de N.

Assoliment objectiu 2: La implementació completa simula tots els passos de l'algorisme de Shor de manera funcional.

4. Resultats i Validació

4.1 Casos de Prova

S'han executat proves amb diversos valors de N, obtenint els següents resultats:

N	Factors obtinguts	Temps aproximat	Intents necessaris
15	3 × 5	< 1s	1-2
21	3 × 7	< 1s	1-3
35	5 × 7	< 2s	1-3
33	3 × 11	< 2s	1-2

Tots els casos han estat factoritzats correctament, validant la implementació.

4.2 Anàlisi de Rendiment

Nombre de qubits: $n = \lceil 2 \log_2(N) \rceil$

- N = 15 → 8 qubits (256 estats simultanis)
- N = 35 → 12 qubits (4096 estats simultanis)

Taxa d'èxit: L'algorisme troba els factors en 1-3 intents en la majoria de casos, consistent amb la probabilitat teòrica d'èxit superior al 50% per intent.

Limitació principal: La simulació clàssica requereix memòria exponencial $O(2^n)$, limitant la factorització a nombres relativament petits. Un ordinador quàntic real operaria amb molts més qubits sense aquest cost prohibitiu.

5. Conclusions

S'han assolit amb èxit tots els objectius plantejats a la pràctica:

✓ **Objectiu 1**: S'han implementat i comprès els conceptes fonamentals de computació quàntica (superposició, entrellaçament, mesura) aplicats a la factorització.

✓ **Objectiu 2**: La simulació implementada reprodueix fidelment tots els passos de l'algorisme de Shor de manera funcional.

✓ **Objectiu 3**: S'ha demostrat teòrica i experimentalment la relació entre la Transformada de Fourier Quàntica i el càlcul del període d'una funció modular.

✓ **Objectiu 4**: La integració de la part clàssica amb el mòdul quàntic permet obtenir correctament els factors de $N = p \cdot q$.

Aportacions Principals

1. **Simulació fidel**: Reprodueix el comportament d'un ordinador quàntic utilitzant tècniques clàssiques, permetent comprendre el funcionament intern de l'algorisme.

2. **Modularitat**: Cada component quàntic està implementat de forma independent i ben documentada, facilitant la comprensió i possibles extensions futures.
3. **Validació empírica**: Els resultats obtinguts confirmen la teoria i validen la correctesa de la implementació per diversos casos de prova.

Aquesta pràctica ha proporcionat una comprensió profunda dels fonaments de la computació quàntica i ha demostrat el potencial revolucionari dels ordinadors quàntics en àmbits com la criptografia, exemplificant com problemes computacionalment intractables per ordinadors clàssics poden ser resolts eficientment amb recursos quàntics.