Complexity Classes: Classical and Quantum

Classical Complexity Classes

P (Polynomial Time)

Definition: Problems solvable in polynomial time by a deterministic Turing machine.
Meaning: Time complexity is $O(n^k)$ for some constant k. These problems are considered efficiently solvable.
Examples:

- Sorting a list (e.g., Merge Sort)
- Shortest path (Dijkstra's algorithm)
- Multiplying numbers
- Primality testing (AKS algorithm)

Analogy: Solving a puzzle that grows slowly in complexity as pieces are added.

NP (Nondeterministic Polynomial Time)

Definition: Problems for which solutions can be verified in polynomial time.
Meaning: Finding a solution might be hard, but verifying a solution is easy.
Examples:

- SAT (Boolean satisfiability)
- TSP (Traveling Salesman Problem)
- Integer factorization

Analogy: Like receiving a completed jigsaw puzzle and checking it's correct.
Note: $P \subseteq NP$, but it's unknown whether $P = NP$.

BPP (Bounded-error Probabilistic Polynomial Time)

Definition: Solvable in polynomial time using randomness, with error probability $< \frac{1}{3}$.
Meaning: Algorithms use coin flips to find answers quickly, but with a small chance of error.
Examples:

- Miller-Rabin primality test
- Monte Carlo simulations

Analogy: A smart friend using intuition and luck, but usually correct.

PSPACE (Polynomial Space)

Definition: Solvable using polynomial memory, regardless of time used.
Meaning: Even if solving takes long, memory usage stays small.
Examples:

- General logic-based games (e.g., Go, Chess with fixed size)
- QBF (Quantified Boolean Formula)

Analogy: Solving a long math problem using a small notebook.
Hierarchy: $P \subseteq NP \subseteq PSPACE$

EXP (Exponential Time)

Definition: Problems solvable in time $O(2^n)$ or worse.
Meaning: Time doubles with each slight increase in input.
Examples:

- Brute-force search
- Solving chess from scratch
- Enumerating all SAT assignments

Analogy: A puzzle that doubles in size with each new piece.

Quantum Complexity Classes

BQP (Bounded-error Quantum Polynomial Time)

Definition: Problems solvable by quantum computers in polynomial time with error $< \frac{1}{3}$.
Meaning: Quantum parallelism allows solving problems more efficiently than classical algorithms.
Examples:

- Shor's Algorithm (factoring)
- Grover's Algorithm (database search)

Analogy: A quantum detective who explores all possibilities at once.
Relation: $BPP \subseteq BQP$

QMA (Quantum Merlin-Arthur)

Definition: Quantum analog of NP: a quantum verifier checks a quantum proof.
Meaning: Verifying quantum states that encode solutions.
Example: Local Hamiltonian Problem (quantum version of SAT).
Analogy: A wizard hands over a magical quantum object that only a quantum machine can verify.

Class Relationships

$$P \subseteq BPP \subseteq BQP \subseteq QMA \subseteq PSPACE \subseteq EXP$$

- P problems are efficiently solvable classically.
- NP includes problems that are easy to verify.
- BPP adds randomness; BQP adds quantum power.
- QMA extends NP to quantum settings.
- $PSPACE$ covers space-efficient problems, regardless of time.
- EXP contains very hard, exponential-time problems.

Open Questions

- Is $P = NP$?
- Is $BQP \subseteq NP$?
- Does $NP = PSPACE$?

Summary Table

Class	Type	Model	Examples	Efficient?
P	Classical	Deterministic	Sorting, Dijkstra	Yes
NP	Classical	Verifiable	SAT, TSP, Factoring	Maybe
BPP	Classical	Randomized	Primality Testing	Yes (probabilistic)
PSPACE	Classical	Space-limited	QBF, Logic games	No (time-intensive)
EXP	Classical	Exponential time	Brute-force chess	No
BQP	Quantum	Quantum algorithm	Shor, Grover	Yes (quantum)
QMA	Quantum	Quantum verifier	Local Hamiltonian	Unknown

Quantum Circuit Complexity: $\mathcal{O}(N/\epsilon)$ for Approximating a Unitary

In the context of quantum circuit complexity, we often aim to implement a unitary operator U using a finite set of elementary quantum gates. However, exact implementation is not always possible or practical, so we instead construct an approximation $\tilde{U}$ such that:

$$\|U - \tilde{U}\| \leq \epsilon,$$

where $\epsilon > 0$ is the desired precision and $\|\cdot\|$ denotes an appropriate norm (e.g., the operator norm). The quantum circuit complexity, i.e., the number of gates required to approximate U, may scale as:

$$\mathcal{O}\left(\frac{N}{\epsilon}\right),$$

where:

- N is a parameter representing the problem size (e.g., number of qubits, input size, or number of terms in a Hamiltonian).
- ϵ is the desired approximation precision.

This complexity indicates a trade-off:

- Higher precision (smaller ϵ) requires more quantum gates.
- Larger problem size (N) also increases the number of gates.

Example: Hamiltonian Simulation

For example, in first-order Trotterization of time evolution under a Hamiltonian $H = \sum_{j=1}^{N} H_j$, we approximate:

$$e^{-iHt} \approx \left(\prod_{j=1}^{N} e^{-iH_j t/r}\right)^r,$$

where r is the number of Trotter steps. To achieve precision ϵ, it can be shown that $r = \mathcal{O}(Nt^2/\epsilon)$, leading to a gate complexity of:

$$\mathcal{O}\left(\frac{N}{\epsilon}\right),$$

for fixed time t, assuming each $e^{-iH_j t/r}$ can be implemented efficiently.

Conclusion

Thus, the expression $\mathcal{O}(N/\epsilon)$ captures how both the size of the quantum problem and the required precision influence the overall cost of approximating a unitary operation in a quantum circuit.