Quantum Complexity Theory

CSE 490Q: Quantum Computation

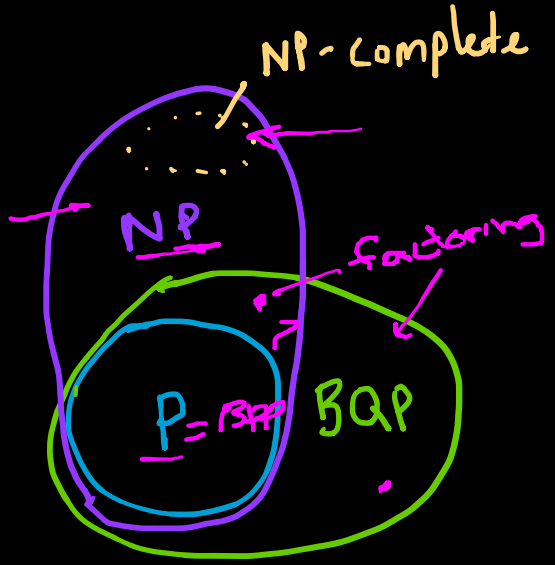
Overview

- We will discuss a key complexity results related to quantum computation
- We will not prove any of these. (Complexity results are difficult, in general.)
- However, the basic ideas are important in order to understand what sorts of outcomes are likely (not) achievable by quantum computers.
- Study of algorithms and complexity pair well together:
  - algorithms tries to show that problems can be solved
  - complexity tries to show that problems cannot be solved
  - (if you can't find an algorithm, maybe look for a complexity result)

Complexity Theory

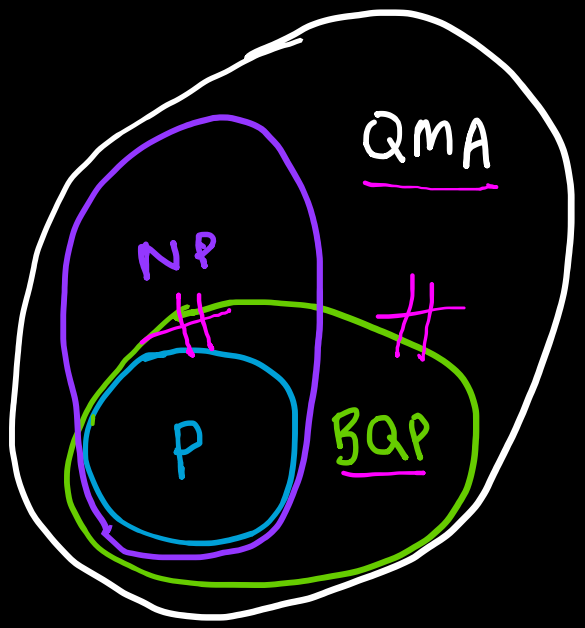
- Organize problems into classes with similar computational difficulty
- Typically restricted to decision (0/1 answer) problems
- Examples we have already seen:
  - P = solvable in polynomial time by a classical computer
  - BPP = solvable in polynomial time with >2/3 success probability by a classical computer with access to random numbers
  - BQP = solvable in polynomial time with >2/3 success probability by a quantum computer

Consensus Views



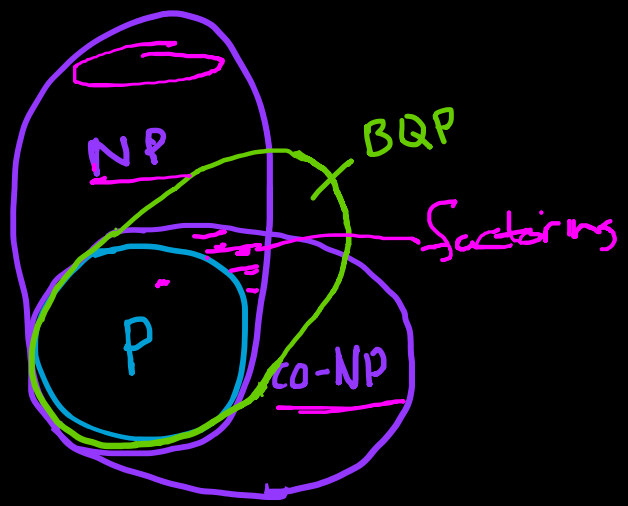
- P ≠ NP is strongly believed but unproven
  - physicists consider this a *natural law*
  - in particular, this means no NP-complete problem can be solved in polynomial time
- P = BPP is strongly believed but unproven
- P ≠ BQP is proven assuming factoring is not in P
  - BQP includes Hamiltonian simulation which is not in NP
- NP ⊄ BQP is strongly believed but unproven
  - physicists consider this a natural law
  - be strongly skeptical of any claim that quantum computers can solve any NP-complete problem

Consensus Views



- QMA is the quantum analogue of NP
  - stands for ”quantum Merlin-Arthur”
  - Merlin is a wizard who can solve hard problems for you, but Arthur, a mere human, still must be able to verify it
- BQP ≠ QMA is strongly believed
  - implied by P ≠ NP or P ≠ BQP

Heuristics



- When the result is 1, NP requires that to be verifiable.
- When the result is 0, co-NP requires that to be verifiable.

- Interestingly, BQP includes a number of interesting problems in the class NP ⋂ co-NP
  - this includes factoring

- These classes also relate to "statistical zero-knowledge"

- Lots of open and interesting problems…
  - these are probably all extremely difficult though ☺

Quantum Speedups

- Shor's algorithm for factoring achieves an exponential speedup
- Grover's algorithm achieves only a quadratic speedup

- When should we expect exponential versus quadratic (versus none)?
  - some standard results shed light on this…

Quantum Query Complexity

- Saw problems like Deutsch-Josza's where we can prove that no classical algorithm can match the same performance
- That problem gave the input in the form of an oracle (a function f)
  - if f is provided as a circuit, we don't know how to prove that a classical algorithm couldn't somehow solve the problem by studying it closely
- We also measured the running time in terms of function calls (queries)
  - complexity theory using this measure of time is called query complexity
  - in principle, it allows exponential time if only polynomially many queries
  - however, it makes provable results possible

Quantum Query Complexity

- As usual, we restrict to decision problems
- Deutsch-Josza is already a decision problem (0 = constant, 1 = balanced)
- Grover can be converted to a decision problem by asking us to distinguish whether f = 0 everywhere or if there is some x such that f(x) = 1
  - algorithm: apply QPE using the Grover iteration (G)
  - if the phase angle is 0 (i.e, not rotating), then f = 0 everywhere
- "Simon's problem" is another historically important example
  - (too similar to others we looked at)

Quantum Query Complexity

- Key property is whether any function f is allowed
  - Deutsch-Josza restricts to only f is constant or balanced — exponential speedup
  - Grover (decision version) allows any function f — polynomial speedup

- It is known that this is true in general:
  - quantum speedups are at most polynomial if they allow any function f

Quantum Query Complexity

- Beals et. al (1998) proved the following:
  - if there is a quantum algorithm solving the problem with T queries,
  - then there is a classical algorithm that solves it with $O(T^6)$ queries

- Furthermore, they showed a bound of $O(T^2)$ when answer to the problem is unchanged when all the input bits to f are permuted
  - e.g., replace f(a,b,c) with g(a,b,c) = f(b,c,a)

- This is true of Grover's algorithm, so a quadratic speedup is optimal
  - permuting the bits just changes which x satisfies f(x) = 1, but it does not change whether f = 0 everywhere or not

Quantum Query Complexity

- Aaronson et al. (2020) improved the general case to $O(T^4)$
  - this and other results rely on Boolean Fourier analysis
    - not too hard, but it would take us too far afield to go there…
  - Aaronson's result relies on the recent breakthrough of Hao Huang, which proved the sensitivity conjecture for Boolean functions

- This is known to be optimal
  - there are problems for which a quartic speedup is achieved by some quantum algorithm

Exponential Speedups

- Without restrictions on the allowed functions, there cannot be an exponential speedup
  - it is also not true, though, that *any* restriction at all makes an exponential speedup possible
- Intuition is that quantum algorithms need to take advantage of some sort of special structure of the functions
- It could be, e.g., that f needs to have special algebraic properties
  - the Hidden Subgroup Problem (coming later) is an example
- Interesting to note that the QFT is a key ingredient in successful examples