Introduction to Quantum Information Processing

Lecture 19

Richard Cleve

Overview of Lecture 19

- Approximately universal sets of gates
- More on complexity classes
  - **NP**: definitions and examples of problems therein
  - **FACTORING** versus **NP** and **co-NP**
  - quantum speed-up for **NP**-complete problems
- Optimality of Grover's search algorithm

***approximately* universal sets of gates**

A universal set of gates

Theorem: any unitary operation U acting on k qubits can be decomposed into $O(4^k)$ CNOT and one-qubit gates

Thus, the set of ***all*** one-qubit gates and the CNOT gate are ***universal*** in that they can simulate any other gate set

Question: is there a ***finite*** set of gates that is universal?

Answer 1: strictly speaking, ***no***, because this results in only countably many quantum circuits, whereas there are uncountably many unitary operations on k qubits (for any k)

Approximately universal gate sets

Answer 2: ***yes***, for universality in an ***approximate*** sense

As an illustrative example, any rotation can be approximated within any precision by repeatedly applying

$$R = \begin{pmatrix} \cos(\sqrt{2}\pi) & -\sin(\sqrt{2}\pi) \\ \sin(\sqrt{2}\pi) & \cos(\sqrt{2}\pi) \end{pmatrix}$$

some number of times

In this sense, R is ***approximately universal*** for the set of all one-qubit rotations: any rotation S can be approximated within precision ε by applying R a suitable number of times

It turns out that $O\left((1/\varepsilon)^c\right)$ times suffices (for a constant c)

Approximately universal gate sets

Theorem: the gates CNOT, H, and $S = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$

are ***approximately universal***, in the sense that any unitary operation on k qubits can be simulated within precision ε by applying $O\left(4^k \log^c(1/\varepsilon)\right)$ of them (c is a constant)

[Solovay '96][Kitaev '97]

more on complexity classes

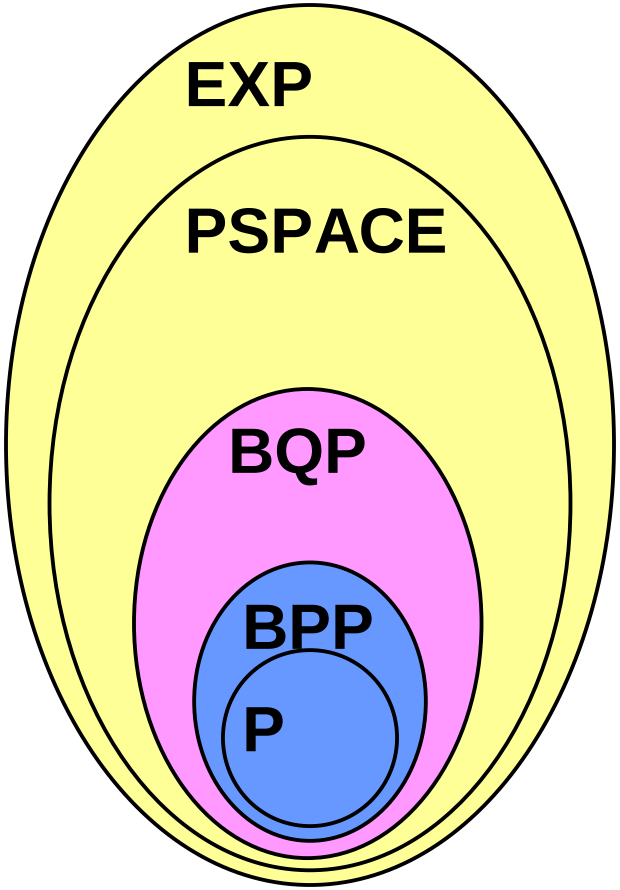
Complexity classes

Recall from Lecture 6:

- **P (polynomial time):** problems solved by $O(n^c)$-size classical circuits (decision problems and uniform circuit families)
- **BPP (bounded error probabilistic polynomial time):** problems solved by $O(n^c)$-size ***probabilistic*** circuits that err with probability $\leq$ ¼
- **BQP (bounded error quantum polynomial time):** problems solved by $O(n^c)$-size ***probabilistic*** circuits that err with probability $\leq$ ¼
- **PSPACE (polynomial space):** problems solved by algorithms that use $O(n^c)$ memory.

Summary of previous containments

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



We now consider further structure between **P** and **PSPACE**

Technically, we will restrict our attention to ***languages*** (essentially $\{0,1\}$-problems)

Many problems of interest can be cast in terms of languages

For example,

FACTORING = $\{(x,y) : \exists\, 2 \leq z \leq y$, such that z divides $x\}$

NP

Define **NP (non-deterministic polynomial time)** as the class of languages whose ***positive*** instances have "witnesses" that can be verified in polynomial time

Example: Let **3-CNF-SAT** be the language consisting of all **3-CNF** formulas that are satisfiable

3-CNF formula:

$$f(x_1,\ldots,x_n) = (x_1 \vee \bar{x}_3 \vee x_4) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_5) \wedge \cdots \wedge (\bar{x}_1 \vee x_5 \vee \bar{x}_n)$$

$f(x_1,\ldots,x_n)$ is ***satisfiable*** iff there exists $b_1,\ldots,b_n \in \{0,1\}$ such that $f(b_1,\ldots,b_n) = 1$

No sub-exponential-time algorithm is known for **3-CNF-SAT**

But poly-time verifiable witnesses exist (namely, $b_1, \ldots, b_n$)

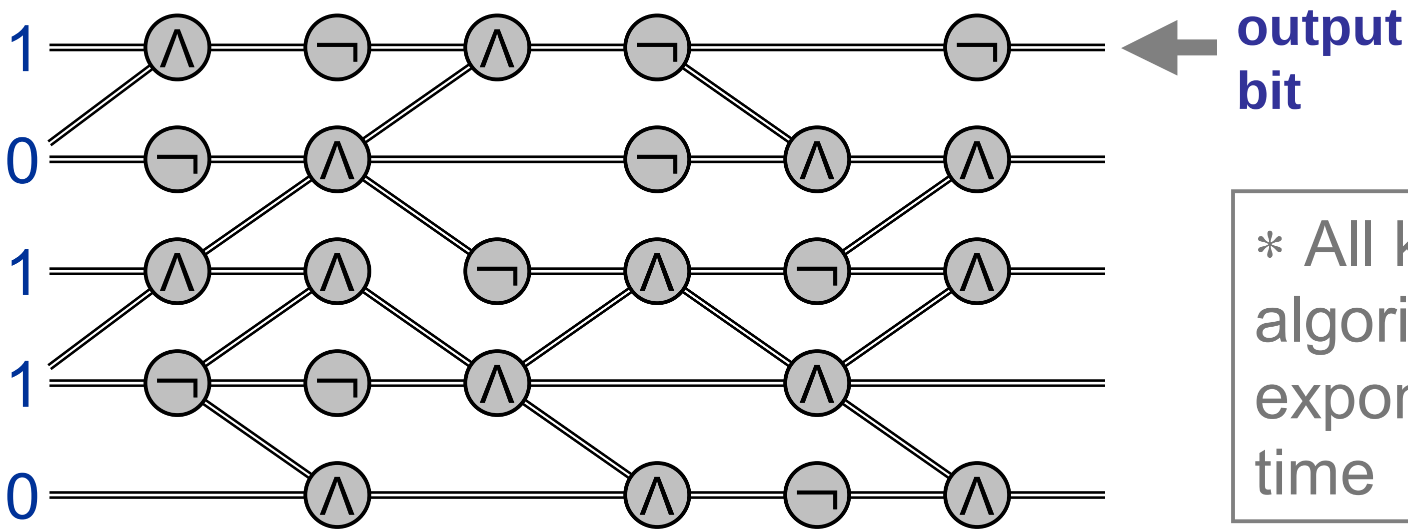
Other "logic" problems in NP

- ***k*-DNF-SAT**:

$$f(x_1,\ldots,x_n) = (x_1 \wedge \bar{x}_3 \wedge x_4) \vee (\bar{x}_2 \wedge x_3 \wedge \bar{x}_5) \vee \cdots \vee (\bar{x}_1 \wedge x_5 \wedge \bar{x}_n)$$

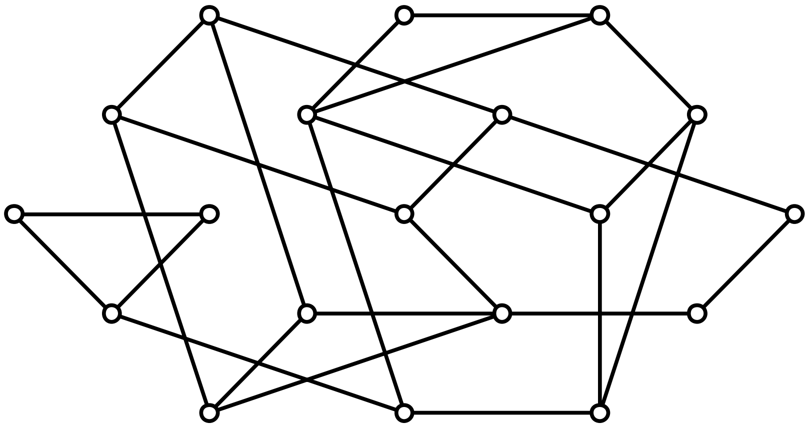
> ∗ But, unlike with *k*-**CNF-SAT**, this one is known to be in **P**

- **CIRCUIT-SAT**:



> ∗ All known algorithms exponential-time

"Graph theory" problems in NP



- **k-COLOR**: does G have a k-***coloring***?
- **k-CLIQUE**: does G have a ***clique*** of size k?
- **HAM-PATH**: does G have a ***Hamiltonian path***?
- **EUL-PATH**: does G have an ***Eulerian path***?

“Arithmetic” problems in NP

- **FACTORING** = $\{(x, y) : \exists 2 \leq z \leq y$, such that z divides $x\}$

- **SUBSET-SUM**: given integers $x_1, x_2, \ldots, x_n, y$, do there exist $i_1, i_2, \ldots, i_k \in \{1, 2, \ldots, n\}$ such that $x_{i_1} + x_{i_2} + \ldots + x_{i_k} = y$?

- **INTEGER-LINEAR-PROGRAMMING**: linear programming where one seeks an ***integer-valued*** solution (its existence)

P vs. NP

All of the aforementioned problems have the property that they ***reduce*** to **3-CNF-SAT**, in the sense that a polynomial-time algorithm for **3-CNF-SAT** can be converted into a poly-time algorithm for the problem

Example:

algorithm for **3-COLOR**

algorithm for **3-CNF-SAT**

If a polynomial-time algorithm is discovered for **3-CNF-SAT** then there is a polynomial-time algorithm for **3-COLOR**

In fact, this holds for ***any*** problem **X** $\in$ **NP**, hence **3-CNF-SAT is *NP*-*hard*** ... and so are **CIRCUIT-SAT**, k-**COLOR**, ...

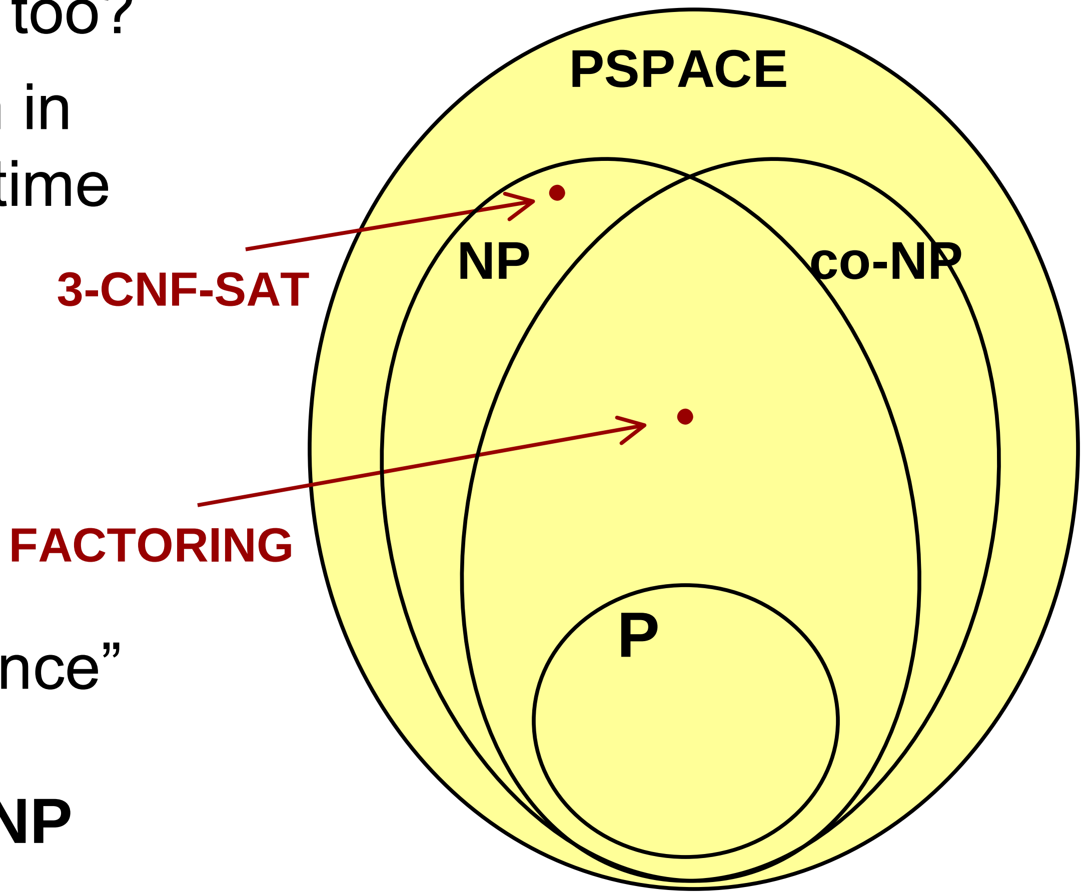
FACTORING vs. NP

Is **FACTORING NP**-hard too?

If so, then ***every*** problem in **NP** is solvable by a poly-time quantum algorithm!

But **FACTORING** has not been shown to be **NP**-hard

Moreover, there is "evidence" that it is not **NP**-hard: **FACTORING** $\in$ **NP**$\cap$**co-NP**

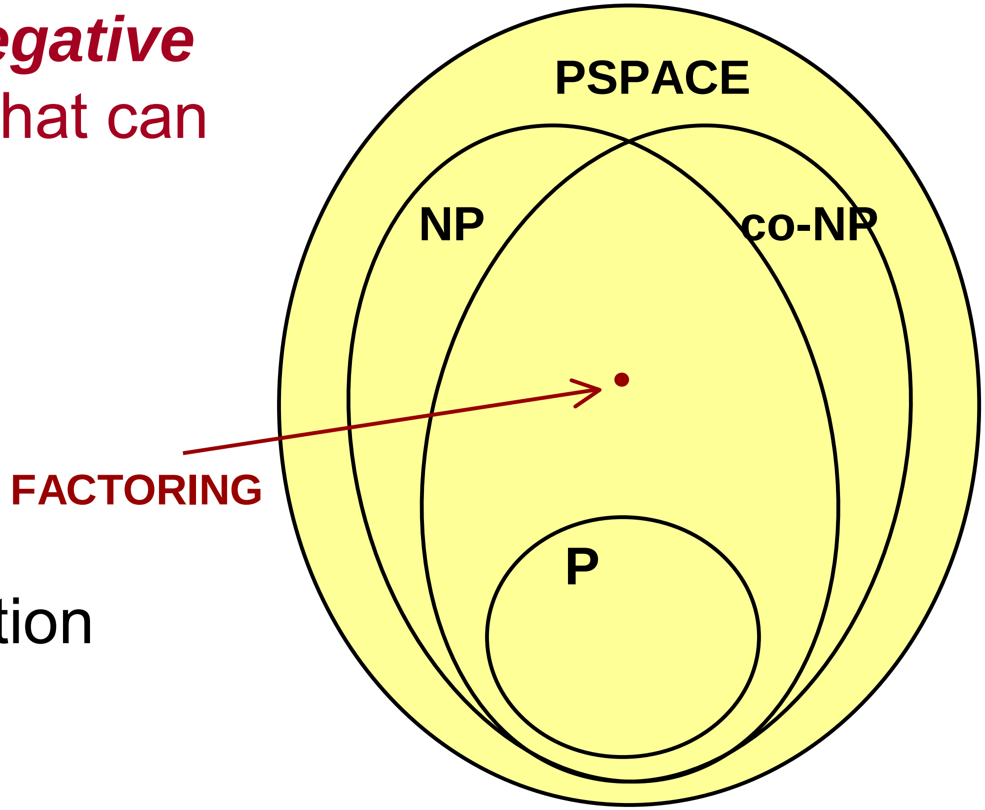


If **FACTORING** is **NP**-hard then **NP** = **co-NP**

FACTORING vs. co-NP

FACTORING = $\{(x, y) : \exists\, 2 \leq z \leq y, \text{ s.t. } z \text{ divides } x\}$

co-NP: languages whose ***negative*** instances have "witnesses" that can be verified in poly-time



Question: what is a good witness for the negative instances?

Answer: the prime factorization $p_1, p_2, \ldots, p_m$ of x will work

Can verify primality and compare $p_1, p_2, \ldots, p_m$ with y, all in poly-time

Quantum speed-up for NP-complete problems

Can use Grover's quantum search algorithm to find a witness ***quadratically*** faster than with known classical algorithms

Example: for **CIRCUIT-SAT**, best classical algorithm is to search for a satisfying assignment, taking time $O(n^c \, 2^n)$

Quantum algorithm takes time $O(n^c \, 2^{n/2})$

optimality of Grover’s search algorithm

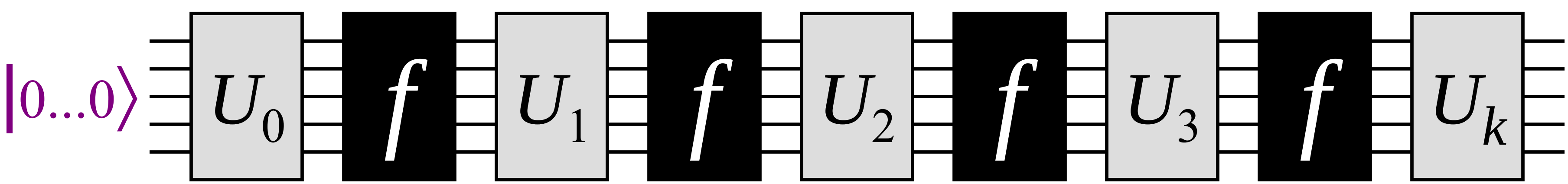
Optimality of Grover's algorithm

Theorem: any quantum search algorithm for $f : \{0,1\}^n \to \{0,1\}$ must make $\Omega(\sqrt{2^n})$ queries to f

Proof (of a slightly simplified version)**:**

Assume queries are of the form

and that a k-query algorithm is of the form

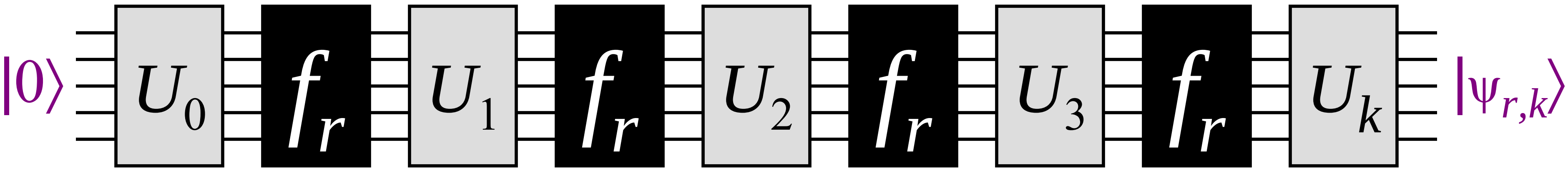


where U_0, U_1, U_2, ..., U_k, are any unitary operations

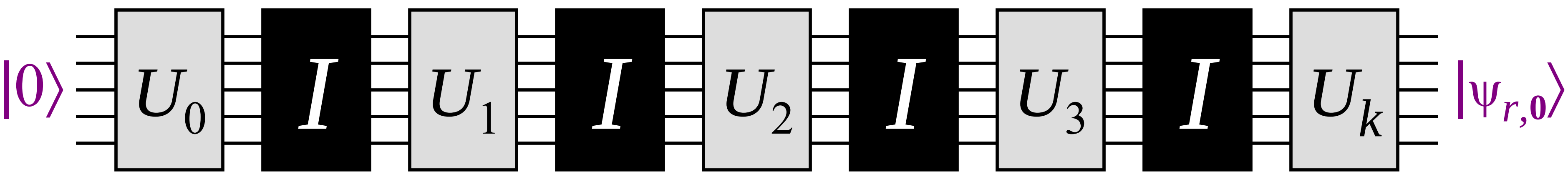
Optimality of Grover's algorithm

Define $f_r : \{0,1\}^n \rightarrow \{0,1\}$ as $f_r(x) = 1$ iff $x = r$

Consider



versus

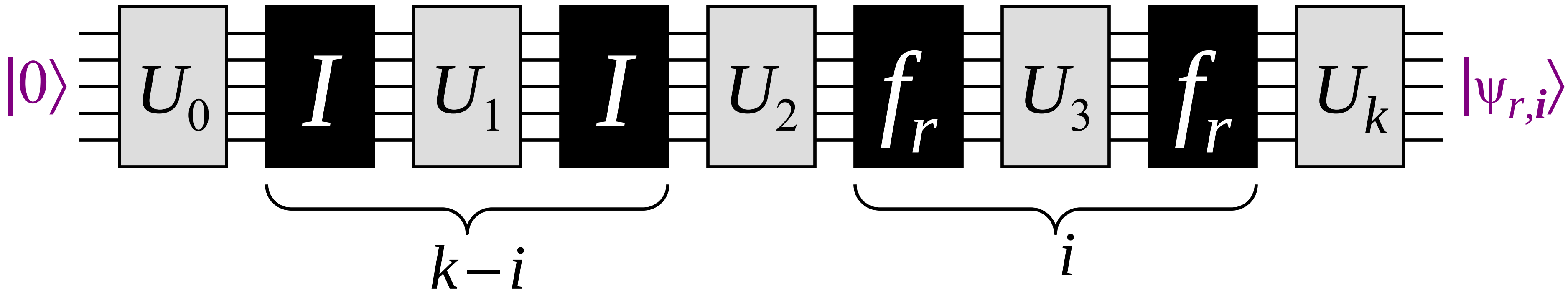


We'll show that, averaging over all $r \in \{0,1\}^n$,

$\| \, |\psi_{r,k}\rangle - |\psi_{r,\mathbf{0}}\rangle \, \| \leq 2k/\sqrt{2^n}$

Optimality of Grover's algorithm

Consider



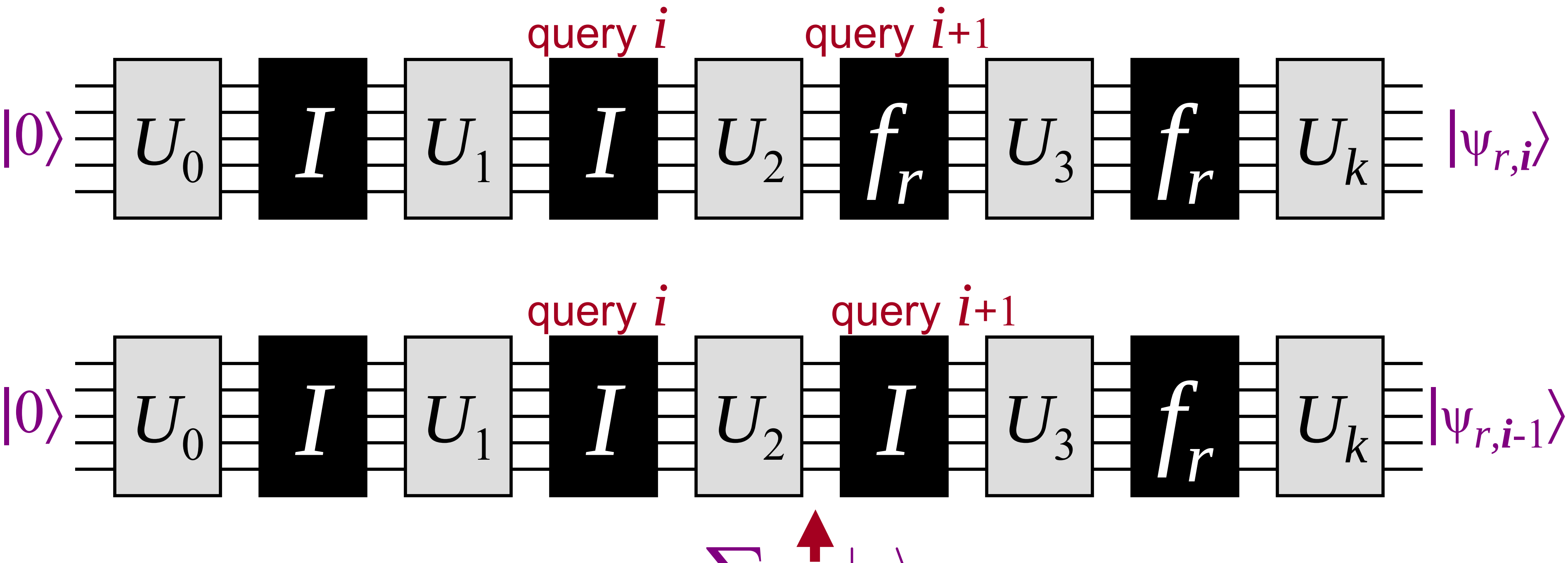
Note that

$$|\psi_{r,k}\rangle - |\psi_{r,\mathbf{0}}\rangle = (|\psi_{r,k}\rangle - |\psi_{r,k-1}\rangle) + (|\psi_{r,k-1}\rangle - |\psi_{r,k-2}\rangle) + \ldots + (|\psi_{r,1}\rangle - |\psi_{r,0}\rangle)$$

which implies

$$\|\, |\psi_{r,k}\rangle - |\psi_{r,\mathbf{0}}\rangle \,\| \leq \|\, |\psi_{r,k}\rangle - |\psi_{r,k-1}\rangle \,\| + \ldots + \|\, |\psi_{r,1}\rangle - |\psi_{r,0}\rangle \,\|$$

Optimality of Grover's algorithm



$$\sum_x \alpha_{i,x} | x \rangle$$

$\| \, |\psi_{r,\mathbf{i}}\rangle - |\psi_{r,\mathbf{i}\text{-}1}\rangle \, \| = |2\alpha_{i,r}|$, since query only negates $|r\rangle$

Therefore, $\| \, |\psi_{r,k}\rangle - |\psi_{r,\mathbf{0}}\rangle \, \| \leq \sum_{i=0}^{k-1} 2|\alpha_{i,r}|$

Optimality of Grover's algorithm

Now, averaging over all $r \in \{0,1\}^n$,

$$\frac{1}{2^n}\sum_r \left\| |\psi_{r,k}\rangle - |\psi_{r,0}\rangle \right\| \le \frac{1}{2^n}\sum_r \left(\sum_{i=0}^{k-1} 2|\alpha_{i,r}| \right)$$

$$= \frac{1}{2^n}\sum_{i=0}^{k-1} 2\left(\sum_r |\alpha_{i,r}| \right)$$

$$\le \frac{1}{2^n}\sum_{i=0}^{k-1} 2\left(\sqrt{2^n}\right) \qquad \text{(By Cauchy-Schwarz)}$$

$$= \frac{2k}{\sqrt{2^n}}$$

Therefore, for ***some*** $r \in \{0,1\}^n$, the number of queries k must be $\Omega\left(\sqrt{2^n}\right)$, in order to distinguish f_r from the all-zero function

This completes the proof

THE END