Informal Review

Research Background (General Understanding)

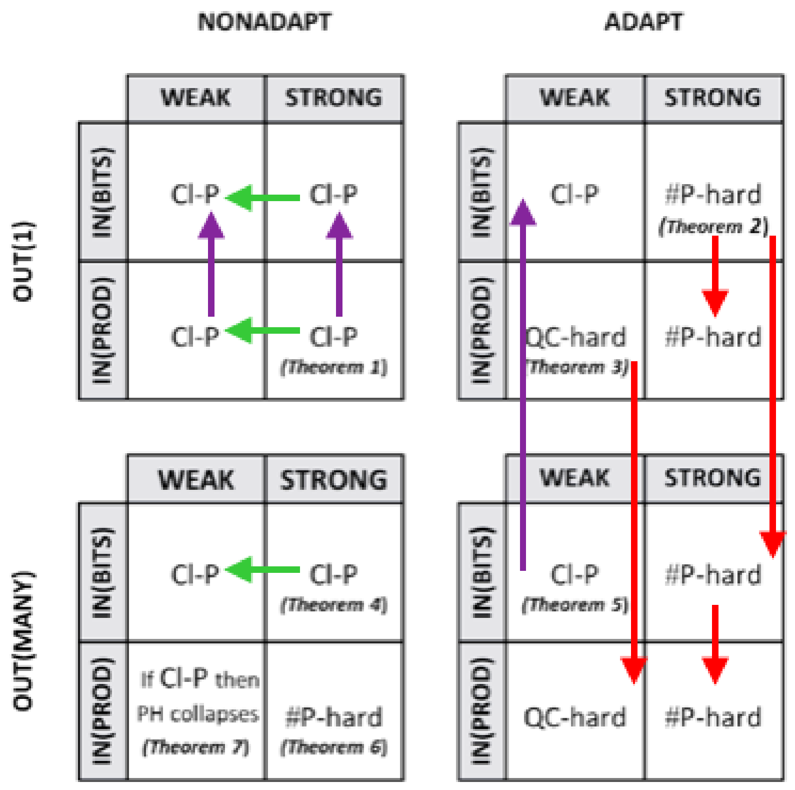
In this section, I would like to briefly talk about what is the underlying question this paper would like to address (at least from my basic understanding). In essence, it discusses about what class(es) of quantum algorithms could be efficiently (i.e. polynomial time complexity) simulated on a classical computer.

In Part II QIC course, we have learned several quantum algorithms which can achieve a speed-up over their classical counterparts, with some even crossing the polynomial vs exponential divide. However, one of the primary reasons why these quantum algorithms are so efficient is due to the fact that we are counting the number of operations in a somehow "dodgy" and "mysterious" way (at least from the perspective of someone who has very little knowledge in the field and does not know how quantum computers work physically... such as me...). In quantum algorithms, one query to the oracle is counted as one operation, and so is one unitary action. I don't exactly know how much time (in terms of time complexity) it actually takes for a quantum computer to run one query or one unitary action, but I am sure it definitely takes many classical bit operations to simulate one query to the oracle or one unitary action. Now, the key problem is, how many is that "many"? When we classically simulate a quantum algorithm, will its efficiency still be preserved? (If the answer is "yes", then we don't even need a quantum computer to manifest the power of quantum algorithms.)

Before investigating on the efficiency of classical simulation, our first question is, how can we classically simulate a quantum algorithm which consists of Unitary, Measure and Ancilla as its basic operations? Traditionally, we restrict our attention to the qubits which could be stabilised by operations from the Pauli group, unitary gates which fall within the normaliser of Pauli group (Clifford operations) and measurements of observables in Pauli group. The result of classical simulation of quantum algorithms with these restrictions is the well-known Gottesman-Knill Theorem. The proof of Gottesman-Knill Theorem is based on the idea that "classical simulation is performed by simply keeping track of the generator of the stabiliser".

This paper considers "Clifford computations with a variety of additional ingredients" and discusses about the complexity of respective classical simulations.

Research Result



← : If a set of tasks is CI-P, then any subset is CI-P, too.

↓ : If a subset of tasks is QC- or #P-hard, then the full set is QC- or #P-hard.

↑ : If strong simulation is CI-P, then weak simulation is CI-P, too.

Summary of Proofs

Basic Set-up: Two ways of representing Pauli gates

1) Any Pauli operator P can be written uniquely as $P = \alpha X(\underline{a})Z(\underline{b})$ for some $\alpha = \pm 1, \pm i$, $\underline{a} = a_1 ... a_\mathrm{n}$, $\underline{b} = b_1 ... b_\mathrm{n}$. Hence, any P could be represented by a $(2n+2)$-bit string. When we apply a basic Clifford gate to P by conjugation, we are just changing the representation of P, and it takes $O(n)$ time.

2) Any Pauli operator P could be written as $P = \gamma P_1 \otimes ... \otimes P_\mathrm{n}$.

Theorem 1: NONADAPT, IN(PROD), OUT(1) and STRONG: Cl-P

• Non-adaptive Clifford circuit C may be assumed to be unitary.

• Use the trick that $p_0 - p_1 = \langle\beta|Z|\beta\rangle = \langle\alpha|C^\dagger ZC|\alpha\rangle$, where α and β are single qubits and $C^\dagger ZC$ is a Pauli operator. Generalise to the case where α and β and n-qubits.

Theorem 2: ADAPT, IN(BITS), OUT(1) and STRONG: # P-hard

• With the availability of adaptation and input in computational basis states, we can apply the Toffoli gate. With the availability of computational basis state inputs, using X and Toffoli construction give the universal classical computation.

• Generate a uniformly random n-qubit computational basis state $|x\rangle$, then apply $A_\mathrm{f} : |x\rangle|0\rangle \to |x\rangle|f(x)\rangle$ and finally measure the qubit line of $|f(x)\rangle$ to give a single bit output. The probability of obtaining 1 is $\#f/2^n$, so strong simulation is # P-hard.

Theorem 3: ADAPT, IN(PROD), OUT(1) and WEAK: QC-hard

• With the availability of adaptation, the gate CX is available.

• From other papers, it is known that Clifford gates with the addition of the phase gate $S = diag(1, e^{i\pi/4})$ are sufficient for universal quantum computation.

• $M_\mathrm{a}(x)CX_\mathrm{ai}|\psi\rangle|\frac{\pi}{4}\rangle_\mathrm{a}$ results in $S_\mathrm{i}|\psi\rangle|0\rangle_\mathrm{a}$ if $x=0$ and $e^{i\pi/4}S_\mathrm{i}{}^{-1}|\psi\rangle|1\rangle_\mathrm{a}$ if $x=1$. Apply the Clifford gate $T = S^2$ to line i when $x=1$.

Theorem 4: NONADAPT, IN(BITS), OUT(MANY) and STRONG: Cl-P

• For nonadaptive circuit, the circuit can be assumed to be unitary and wlog, the basis input and output are both sequences of zeros. Let $p = prob(0...0)$ be the probability of obtaining 0 from measurement of each of the lines 1 to m. Using $|0\rangle\langle 0| = (I+Z)/2$ and writing $\mathbf{t} = t_1 ... t_\mathrm{m}$ for m-bit strings we have $p = \frac{1}{2^m}\sum_\mathbf{t}\langle 0^n|C^\dagger Z(\mathbf{t})C|0^n\rangle$. By the first representation of Pauli gates, for each $\mathbf{t}$, the $n-qubit$ Pauli operators can be computed efficiently through its labels.

• Using the fact that $\langle 0|X|0\rangle = 0$ and $\langle 0|Z|0\rangle = \langle 0|I|0\rangle = 1$, we have $p = \frac{1}{2^m}\sum_{\mathbf{t}\in T_0}(-1)^{u(\mathbf{t})}$, where $T_0 = \{\mathbf{t} : \mathbf{a(t)} = 0^n\}$. We can compute a basis $\{c_1, ..., c_\mathrm{l}\}$ of T_0 in poly(n) time. $p = \frac{1}{2^m}\sum_\mathbf{s}(-1)^{u(\sum s_\mathrm{i}c_\mathrm{i})} = \frac{1}{2^m}\sum_\mathbf{s}(-1)^{\mathbf{k\cdot s}} \neq 0$ iff $k = 0^l$, where $u(c_\mathrm{i}) = k_\mathrm{i}$.

Theorem 5: ADAPT, IN(BITS), OUT(MANY) and WEAK: Cl-P

- Borrowing the idea from the proof of Theorem 4.
- ? see the first question in Confusing Parts.

Theorem 6: NONADAPT, IN(PROD), OUT(MANY) and STRONG: # P-hard

• Efficient strong simulation of A would imply efficient strong simulation of universal quantum computation: Let D be any quantum circuit comprising basic Clifford gates and S gates with a product state input and single bit output denoted y. Suppose there are K S gates in D. For each such gate introduce an ancilla in state $|\frac{\pi}{4}\rangle$ and replace the S gate by the sequence of operations as in the proof of Theorem 3, resulting in a non-adaptive circuit D' now involving only basic Clifford gates.

• D' has $K+1$ outputs viz. y and measurements of the K ancilla lines denoted $a_1, ..., a_K$, and we have $Prob_{\mathrm{D}}(y) = Prob_{\mathrm{D'}}(y|0_{a_1}, ..., 0_{a_K}) = Prob_{\mathrm{D'}}(y0_{a_1}, ..., 0_{a_K})/Prob_{\mathrm{D'}}(0_{a_1}, ..., 0_{a_K})$. Both D' probabilities in the quotient could be computed from the strong simulation of D.

• Efficient strong simulation of universal quantum computation would provide an efficient solution of the # SAT problem: using the ideas described in the proof of Theorem 2.

Theorem 7: NONADAPT, IN(PROD), OUT(MANY) and WEAK: collapse of PH

?

Confusing Parts

1) In the elementary simplification of circuit structures, why do we choose CX gate to replace the intermediate measurement? Why does it work?

2) Is there a relationship between QC-hard and # P-hard?

3) I sort of understand the basic definitions of weak and strong simulations, but I am not sure how their features are related to the way they are being classically simulated.

4) Don't quite understand the proof of Theorem 7.

Further Thoughts

1) In this paper, it also briefly mentions about other related work "on the classical simulation complexity of various extensions generalisations of Clifford circuits". Hence, I am wondering how these extensions and generalisations are being motivated, i.e. why we choose certain extensions and generalisations instead of the others.

2) Although we have shown that certain classes of quantum algorithms have efficient classical simulation, I am not sure what kind of classical problems these classes of quantum algorithms can solve. I am wondering if there exists any decision problem with (traditionally-known) inefficient classical algorithms, but efficient quantum algorithms which belong to the class of Cl-P. Probably it doesn't exist... It seems to me that current research on the classical simulation complexity is a bit disjoint from research on the quantum algorithms which perform superior to their classical counterparts. It might be helpful to draw the results from both fields together to gain further insights.

3) I felt very amazed at the stabiliser formalism which provided the foundation of Gottesman-Knill Theorem and various other classical techniques. Hence, I am wondering if there are any other structure/insights which we can borrow from Pure Mathematics concepts to formalise a larger/different set of quantum operations. (This sounds much harder than the first two ones...)