

NISQ machine의 성능 향상을 위한 Qiskit 기반 quantum error mitigation 적용 연구

김보선, 손일권, 배광일, 이원혁

한국과학기술정보연구원

{boseon12, d2estiny, kibae, livezone}@kisti.re.kr

A Study on the Application of Qiskit-based Quantum Error Mitigation for Improving Performance in NISQ Machines

Boseon Kim, IlKwon Sohn, Kwangil Bae, Wonhyuk Lee

Korea Institute of Science and Technology Information.

요약

쇼어의 소인수분해 알고리즘을 통해 양자 컴퓨터를 통한 알고리즘 연구가 가속화되었고, 여러 기업 또한 양자 컴퓨터에 관한 관심과 연구가 집중되고 있다. 현재 양자 컴퓨터는 노이즈로 인해 성능을 저해하는 오류가 발생하고 있어, 양자 컴퓨터의 오류를 완화시켜 정확한 결과를 도출하는 방법인 error mitigation에 대한 연구를 수행하고 있다. 본 논문은 15큐비트를 사용하는 toy example인 Grover 알고리듬의 노이즈와 error mitigation 여부에 따른 세 가지 실험을 진행하여 각 실험 결과를 바탕으로 노이즈가 있는 환경에서 error mitigation의 성능을 분석하고자 한다.

I. 서 론

쇼어의 소인수분해 알고리즘은 고전 컴퓨터를 사용하는 알고리즘보다 필요한 연산량 n 에 대해 지수함수적으로 적다. 이를 계기로 양자 컴퓨터를 통한 알고리즘 개발이 본격화되었고, Google과 IBM, Amazon, Microsoft 또한 양자 컴퓨터에 관한 관심과 연구가 집중되고 있다. 양자 컴퓨터는 여러 가지 상태로 존재할 수 있는 큐비트의 특성에 의해 오류 가능성도 큐비트가 많아질수록 커진다. 이에 해결 방법인 Error mitigation은 양자 계산에서 노이즈의 영향을 줄이는 것을 목표로 한다. IBM은 지속적인 Error Mitigation 연구를 수행하여, ZNE(Zero-noise extrapolation)와 PEC(Probabilistic error cancellation)를 포함하여 Error mitigation에 대한 다양한 접근 방식을 다루고 있다.

우리는 양자컴퓨팅을 위한 플랫폼인 IBM의 Qiskit을 기반으로 NISQ machine의 성능 향상을 위한 quantum error mitigation 방법을 적용하는 연구를 수행한다. 본 연구는 15 큐비트를 사용하는 toy example의 noise 와 error mitigation의 여부에 따른 실험을 진행하여 각각의 결과를 바탕으로 성능을 분석하고자 한다. 본 논문은 1장 서론에 이어 2장에서 관련 연구를 제시하고 3장에서 error mitigation 적용 방법을 설명한다. 4장에서는 실험 결과를 설명하고 5장에서 결론 및 향후 연구를 제시하는 것으로 본 논문을 마친다.

II. 관련 연구

Mitiq은 양자 컴퓨터에서 Error mitigation을 구현하기 위한 파이썬 툴킷으로 양자 프로그램을 컴파일하여 소프트웨어 수준에서 문제점을 줄이기 위한 방법으로 Error mitigation을 수행한다. Mitiq의 Error mitigation 기술은 ZNE(Zero-noise extrapolation), PEC(Probabilistic error cancellation), CDR(Clifford data regression), DDD(Digital dynamical decoupling), REM(Readout-error mitigation)가 있다. Mitiq

을 사용하기 위해 양자 회로를 작성할 수 있는 프로그래밍 언어인 프론트 엔드와 회로를 실행할 수 있는 양자 컴퓨터 혹은 시뮬레이터인 백엔드가 필수적이다. Mitiq에서 지원하는 프론트엔드는 Qiskit, Cirq, pyQuil, Braket, PennyLane이며, 백엔드는 IBMQ, Regetti, IonQ, PyQuil이다.

III. 실험

A. Observable 적용한 expectation values

Qiskit의 Error mitigation 효과를 입증하기 위해 15큐비트를 사용한 Grover 알고리듬에 ZNE를 적용하여 실험한다. 그림 1은 ZNE를 적용하였을 때, number_of_shots(1, 10, 100, 1000)에 따른 expectation_values 값을 나타낸다. Observable은 ['|||||Y']를 사용한다. 그림과 같이 회로 반복 횟수인 number_of_shots이 증가할수록 expectation_values가 0에 수렴하는 것을 알 수 있다.

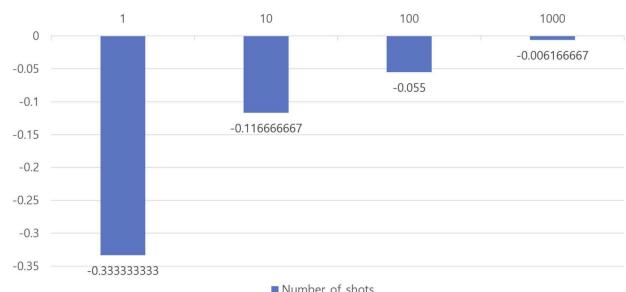


그림 1 Observable 적용한 number_of_shots에 따른 expectation_values

B. Error mitigation 적용

양자 컴퓨터를 활용한 양자 계산에서 노이즈의 영향을 줄이기 위해 Error mitigation 방법을 적용하고자 한다. 우리는 실험을 위해 Qiskit

과 IBMQ를 사용하며, Mitiq 파이썬 룰킷을 사용하여 양자 컴퓨터에서 ZNE 기법을 통한 Error mitigation을 수행한다. Error mitigation 을 적용할 알고리듬은 15큐비트를 사용한 Grover 알고리듬이다.

양자 컴퓨터를 활용한 양자 계산에서 노이즈의 영향을 줄이기 위해 Error mitigation 방법을 적용하고자 한다. 우리는 실험을 위해 Qiskit 과 IBMQ를 사용하며, Mitiq 파이썬 룰킷을 사용하여 양자 컴퓨터에서 ZNE 기법을 통한 Error mitigation을 수행한다. Error mitigation 을 적용할 알고리듬은 15큐비트를 사용한 Grover 알고리듬이다. 노이즈가 없는 환경인 시뮬레이터에서의 Grover 알고리듬 성능과 노이즈 가 있는 환경인 양자 컴퓨터에서의 Grover 알고리듬 성능을 비교하며, 더불어 노이즈가 있는 환경인 양자 컴퓨터에서의 Error mitigation인 ZNE를 적용한 Grover 알고리듬 성능을 함께 비교한다.

노이즈가 있는 환경에서 Error mitigation을 적용한 경우(with error mitigation)와 하지 않은 경우(without error mitigation)을 비교하기 위해 동일한 환경에서 Grover 알고리듬 실험을 진행한다. noise 가 있는 환경에서 ibmq_executor 함수로 qcircuit 회로 인자를 넘겨주어, job을 실행하고 job.result()를 얻는다. job.result()는 job이 실행된 후 즉시 결과를 반환하며, 결과값 통계를 get_counts() 함수를 통해 얻을 수 있고 이를 통해 현재까지 실행했던 실험의 합계된 결과를 확인 할 수 있다. get_counts()는 딕셔너리 형태이며, 딕셔너리에는 이진 형식의 문자열과 회로의 레지스터에 따라 구분된 키를 사용하여 각 큐비트에 대한 카운트가 있다. 예를 들어 result.get_counts(qcircuit)의 결과가 {'0': 498, '1': 502}일 경우, 0과 1은 큐비트의 측정값이고 498과 502는 각 결과가 반복된 횟수이다.

노이즈가 있는 환경에서 Error mitigation을 적용한 Grover 알고리듬을 실행하며, Mitiq의 ZNE를 적용하여 실험한다. Mitiq은 ZNE 방법을 결정하는 Factory를 사용하여 ZNE를 구현하며, 그 중 PolyFactoryMitiq Factory를 사용한다. Factory를 선택한 후, mitiq.zne.zne.execute_with_zne() 함수를 통해 ZNE를 구현한다. Mitiq의 zne.execute_with_zne에 회로 qcircuit과 ibmq_executor 함수, 선택한 factory를 인자로 넘겨주어 job을 실행하여 job.result() 결과를 얻는다.

IV. 실험 결과

세 가지 실험을 각 1000번 수행하여 결과를 도출하였고, 실험 결과를 그래프로 표현한 그림은 그림 2와 같다. x축은 큐비트 측정값이며, y축은 큐비트 측정값이 반복된 횟수이다. 노이즈가 없는 시뮬레이터에서 Grover 알고리듬 실험 결과로 나온 0000, 0111, 1001, 1110 측정값을 제외하고는 오류로 판단한다. 각각의 측정값이 반복된 횟수를 실험마다 비교한다.

A. Noiseless 환경에서 Grover 알고리듬

[('0000', 239), ('0111', 276), ('1001', 234), ('1110', 251)]

B. Noise 환경에서 Grover 알고리듬

[('0000', 53), ('0001', 72), ('0010', 80), ('0011', 67), ('0100', 53), ('0101', 75), ('0110', 64), ('0111', 67), ('1000', 77), ('1001', 73), ('1010', 53), ('1011', 51), ('1100', 49), ('1101', 53), ('1110', 47), ('1111', 66)]

오류가 없는 시뮬레이터와는 다르게 0000, 0111, 1001, 1110 외에 오류로 인해 나올 수 있는 모든 조합이 나온 것을 확인할 수 있다.

C. Noise 환경에서 Error mitigation을 적용한 Grover 알고리듬

[('0000', 71), ('0001', 68), ('0010', 60), ('0011', 84), ('0100', 55),

('0101', 67), ('0110', 66), ('0111', 83), ('1000', 45), ('1001', 64), ('1010', 48), ('1011', 56), ('1100', 41), ('1101', 72), ('1110', 54), ('1111', 66)]

빨간색별은 Error mitigation을 적용하였을 때 올바른 측정값이 검색 된 횟수가 증가한 경우를 표시하였으며, 검정별은 해당 측정값이 검색된 경우가 오히려 감소한 경우를 나타낸다. 결과를 통해 올바른 측정값이 나온 경우는 노이즈가 있는 환경에서 Grover 알고리듬 실험 결과, 0000은 53번, 0111은 67번, 1001은 73번, 1110은 47번이다. 노이즈가 있는 환경에서 Error mitigation을 적용한 Grover 알고리듬 실험 결과, 0000은 71번, 0111은 83번, 1001은 64번, 1110은 54번이다. 이에 따라 Error mitigation 적용을 통해 0000은 18번, 0111은 16번, 1110은 7번 증가하였고, 1001은 7번 감소하였음을 알 수 있다. 노이즈가 있는 환경에서 Error mitigation 을 적용할 경우, 노이즈의 영향을 덜 받아 측정값이 정확하게 도출되는 결과를 볼 수 있다. 이에 따라 성능 향상을 위한 Error mitigation 적용 효과를 확인할 수 있으나, 오히려 감소한 측정값이 한 개 존재한다.

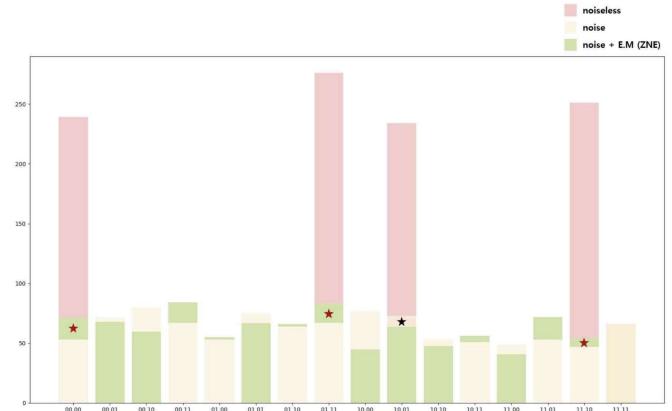


그림 2 Noise와 Error mitigation 여부에 따른 측정값 도출 횟수

V. 결론

양자 컴퓨터의 노이즈 영향을 최소화하기 위한 Error mitigation 연구를 위해 Grover 알고리듬에 노이즈와 Error mitigation 여부에 따른 세 가지 실험을 진행하였다. 노이즈가 있는 환경에서 Error mitigation인 ZNE를 적용할 경우, 올바른 측정값이 도출되는 횟수가 늘어나는 것을 알 수 있었지만 줄어드는 측정값도 존재하였다.

현재 toy example인 Grover 알고리듬에 맞는 Error mitigation을 적용하기 위해 회로에 효율적으로 동작할 수 있는 Error mitigation 기능 구현이 필요하다.

ACKNOWLEDGMENT

본 연구는 2024년도 한국과학기술정보연구원(KISTI)의 기본사업으로 수행된 연구입니다. (K-24-L04-C02)

참 고 문 헌

- [1] LaRose, Ryan, et al. "Mitiq: A software package for error mitigation on noisy quantum computers." *Quantum* 6 (2022): 774.
- [2] Russo, Vincent, et al. "Testing platform-independent quantum error mitigation on noisy quantum computers." *IEEE Transactions on Quantum Engineering* (2023).