Grover's algorithm을 이용한 Graph coloring problem 풀이

신용재(고려대학교), 허준(고려대학교)
tlsdydwo1103@korea.ac.kr, *junheo@korea.ac.kr

Solving Graph coloring problem using Grover's algorithm

Shin Yong Jae, Heo Jun
Korea Univ., *Korea Univ.

요 약

본 논문은 Grover's algorithm의 과정을 살펴보고, 그를 이용하여 기본적인 quantum gate를 통해 Graph coloring problem을 해결하는 회로를 설계한다. 이를 Qiskit에 구현해 시뮬레이션을 진행하여 결과를 확인하고 구현한 회로의 gate 수를 토대로 임의의 Graph를 해결하기 위한 회로에 사용되는 quantum gate의 수를 예측한다.

Ⅰ. 서론

Grover's algorithm은 1996년에 Lov Kumar Grover에 의해 창안된 알고리즘으로 1994년에 창안된 Shor's algorithm 이후 양자 컴퓨팅의 두번째 주요 알고리즘이다. N개의 데이터 중 찾고자 하는 데이터를 찾기 위해서는 모든 데이터를 찾고자 하는 데이터와 비교해 $O(N)$의 복잡도가 필요했던 고전 컴퓨터의 방법과는 달리 Grover's algorithm에서는 양자가 가진 중첩의 특성을 이용하여 같은 데이터를 $O(\sqrt{N})$의 복잡도로 찾을 수 있다. 본 논문에서는 이러한 Grover's algorithm을 이용하여 vertex와 edge로 구성되어 있는 graph에서 인접한 vertex는 서로 다른 color를 칠하게 하는 Graph coloring problem을 해결하는 양자 회로를 Qiskit을 이용하여 설계한 후 시뮬레이션을 진행할 것이다. 또한 구현한 양자 회로의 복잡도를 분석하여 임의의 graph에 대한 복잡도를 예측해 볼 것이다.

Ⅱ. 본론

A. 이론적 배경

1. Grover's algorithm

Grover's algorithm은 다음의 세단계로 구성된다.

1) Initialization

가능한 모든 state들을 동일한 amplitude를 갖도록 중첩한다. 이를 Initialization이라고 하며 중첩된 state들은 $|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$의 식으로 나타난다.

2) Grover iteration

Grover iteration은 앞서 중첩된 state들에서 찾고자 하는 state만의 위상을 변경하는 Quantum oracle U_w와 위상이 변경된 state의 amplitude를 증폭시키는 Diffusion operator U_s를 π/4 √(N/t)회 반복한다. 여기서 N은 가능한 state의 수 즉, 전체 데이터 공간의 크기이고 t는 찾고자 하는 state의 수이다.

Quantum oracle U_w은 중첩된 state들에서 찾고자 하는 state만의 위상을 변경하는 역할을 한다. 이를 식으로 나타내면 $U_w|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle = (-1)^{f(x)}|x\rangle|y\rangle$로 표현할 수 있다. 이때 $f(x)$는 $|x\rangle$가 찾고자 하는 state일 때 1의 값을 가지고 나머지 경우에는 0의 값을 가지는 함수이다.

Diffusion operator U_s는 앞선 U_w에 의해 위상이 변경된 state의 amplitude를 증폭하는 역할을 한다. 이는 $U_s = (2|s\rangle\langle s|-I)$ 로 표현할 수 있다. U_s을 일반적인 state인 $\sum_k a_k|k\rangle$에 적용하면 $(2|s\rangle\langle s|-I)\sum_k a_k|k\rangle = \sum_{k=0}^{N-1}(2\langle a\rangle - a_k)|k\rangle$와 같은 결과가 나온다. 여기서 $\langle a\rangle = \frac{1}{N}\sum_k a_k$이다. 수식의 결과를 보면 state amplitude의 평균에 대한 대칭이 발생하는 것을 확인할 수 있다. 앞서 quantum oracle을 통해 찾고자 하는 state들의 위상을 변경하였는데 찾고자 하는 state의 수는 전체 데이터 공간의 크기와 비교했을 때 소수이므로 이 state들을 평균에 대해 대칭한다면 나머지 state들과 비교했을 때 높은 amplitude를 갖는다.

3) Measurement

Quantum oracle U_w와 Diffusion operator U_s을 반복하면 원하는 state들의 amplitude가 증폭된 상태가 된다. 적절한 반복 횟수는 [2]를 통해 얻고 그 값은 $\frac{\pi}{4}\sqrt{\frac{N}{t}}$이다. 여기에 measurement를 가하여 원하는 state들을 얻는다.

2 Quantum gate

Qiskit에서 Grover's algorithm을 구현하기 위해서는 quantum gate에 대한 정보가 필요하다. 본 논문에서 사용되는 quantum gate는 5가지이고 그들의 역할은 다음과 같다.

1) Hadamard gate

Hadamard gate는 single qubit에 작용하는 gate로 $|0\rangle$을 $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$로 $|1\rangle$을 $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$로 변환한다. Hadamard gate를 거친 qubit은 $|0\rangle$과 $|1\rangle$을 동일한 확률로 가지게 되고 이는 중첩을 의미한다.

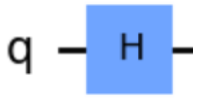

Figure 2. Hadamard gate

2) X gate

X gate도 single qubit에 작용하는 gate로 bit flip을 한다. 즉, $|0\rangle$은 $|1\rangle$로 $|1\rangle$은 $|0\rangle$으로 변환한다.

Figure 3. X gate

3) CX gate

CX gate는 two qubit에 작용하는 gate로 2개의 input qubit을 control qubit, target qubit으로 나누어 control qubit이 $|1\rangle$인 경우에만 target qubit이 flip되고 $|0\rangle$인 경우에는 변하지 않는다.

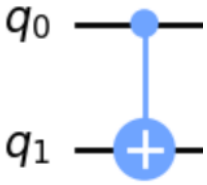

Figure 4. CX gate

4) MCX gate

MCX gate는 Multiple Control X gate로 control qubit을 1개 가졌던 CX gate와 달리 2개 이상의 control qubit을 가지는 gate이다. CX gate와 마찬가지로 control qubit이 모두 $|1\rangle$인 경우에만 target qubit이 flip된다. Figure 4는 control qubit이 2개인 toffoli gate를 나타낸 것이다.

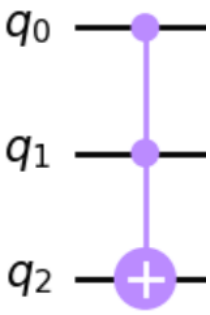

Figure 1. Toffoli gate

5) MCZ gate

MCZ gate는 앞선 MCX에서 target qubit에 X 대신에 Z를 가하는 gate이다. Control qubit이 모두 $|1\rangle$ 인 경우에만 target qubit에 Z가 가해지고 이때 target qubit이 $|0\rangle$이라면 변하지 않고 $|1\rangle$이라면 phase가 바뀌어 $-|1\rangle$가 된다.

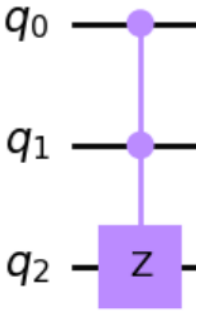

Figure 6. MCZ gate

B. Qiskit에서의 Grover's algorithm 구현

본 논문에서는 2가지 graph에 대한 coloring problem을 해결하기 위해 Grover's algorithm이 구현된 회로를 Qiskit을 이용하여 설계하고 시뮬레이션을 진행한다. 2가지 graph는 다음과 같다.

각 그림의 vertex는 하나의 color를 가지게 되고 그 color는 2 qubit으로 표현된다. Vertex의 color가 정해진다면 해당 vertex와 edge로 연결된 다른 vertex는 서로 다른 color를 가져야한다.

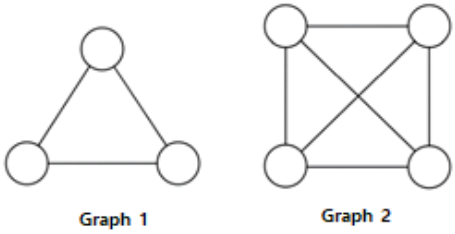

Figure 5. 2가지 graph

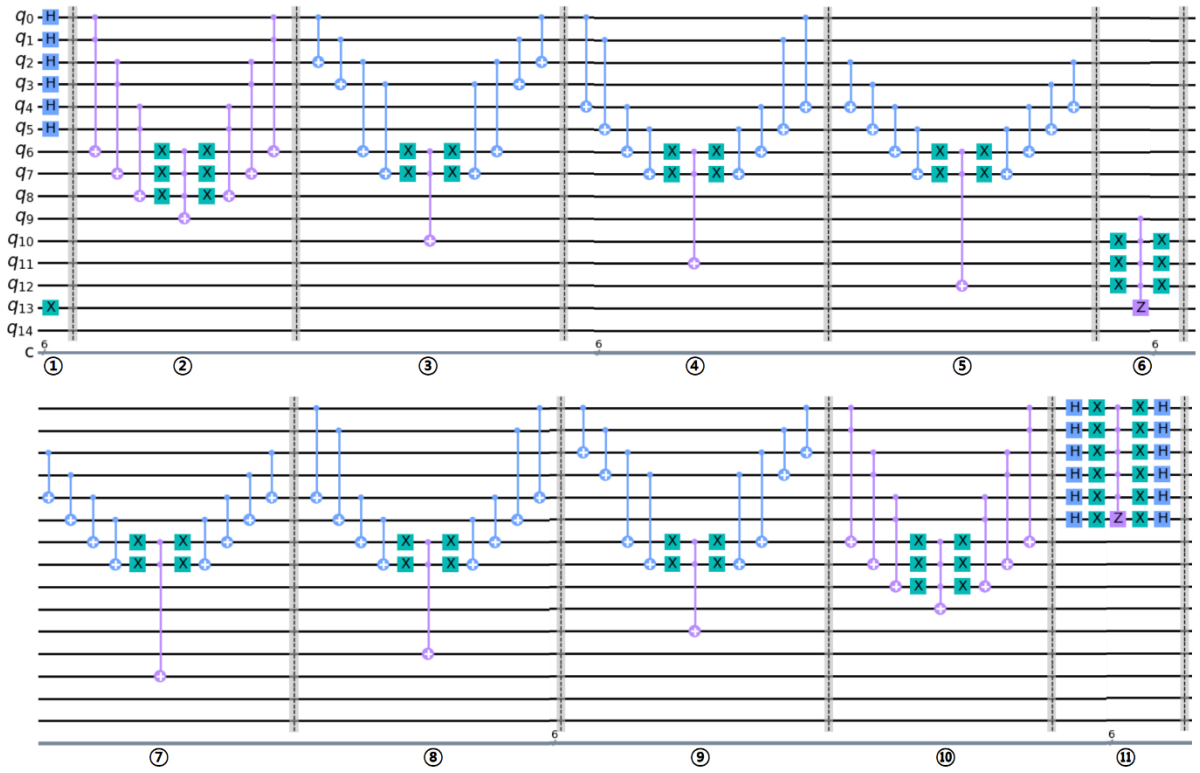

Figure 7. Graph 1 회로

1. Graph 1

먼저 Graph1에 대한 회로는 Figure 7과 같다. 여기서 q0~q5는 각 vertex의 color을 의미한다. 따라서 q0, q1는 첫번째 vertex의 color을 나타내고 q2, q3는 두번째 vertex, q4, q5는 세번째 vertex의 color을 나타낸다. q6~q12는 인접한 2 vertex의 color 비교와 같은 계산의 결과를 저장한다. q13은 q6~q12에 저장된 결과를 종합하여 찾고자 하는 state에 대한 위상을 변경하는데 사용된다. ①은 initialization 에 해당한다. 3 vertex 의 color 을 나타내는 6 개의 qubit 에 Hadamard gate 를 가해준다. 이를 통해 state 들이 중첩된다. ②~⑩은 quantum oracle 에 해당한다. Quantum oracle 은 역할에 따라 세 부분으로 나눌 수 있다. ②는 color 의 수를 3 가지로 줄이기 위해 구성되었다. Graph 1 coloring 에는 3 color 가 필요한데 2 qubit 을 사용한다면 00, 01, 10, 11 의 4 color 가 가능하므로 출력되는 결과의 수가 많아진다. 이를 해결하기 위해 ②가 구성되었고 color 11 을 가지는 state 는 q9 가 $|0\rangle$이 된다. ③은 각 edge 에 연결되어 있는 2 vertex 의 color 를 비교하는 회로로 하나의 vertex 의 color 는 2 qubit 으로 표현되기 때문에 2 vertex 의 2 qubit 을 각각 비교하여 color 가 서로 같은 경우에 q10 가 $|1\rangle$이 된다. 위 graph 는 3 edge 를 가지기 때문에 ④, ⑤에도 같은 회로가 반복된다. ⑥은 ②~⑤에서의 결과를 종합하여 처리하는 회로로 ②는 color 11 을 가지지 않는 경우에만 q9 가 $|1\rangle$이 되고 ③~⑤는 2 vertex 의 color 가 서로 다른 경우에 q10~q12 가 $|0\rangle$이 되므로 q10~q12 에 X gate 를 가해 qubit 의 값을 $|1\rangle$으로 바꾼 뒤 MCZ gate 를 가해 graph 가 color 11 을 가지지 않고 edge 로 연결된 vertex 의 color 가 서로 다른 경우에만 Z gate 가 가해지도록 했다. 이때 MCZ 의 target qubit 인 q13 이 $|1\rangle$인 경우에만 phase 가 바뀌므로 initialization 에서 q13 에 미리 X gate 를 가해준 것을 확인할 수 있다. ⑦~⑩에서는 앞선 과정을 진행하면서 여러 qubit 의 값들이 바뀌었으므로 앞선 과정을 역순으로 진행하여 q7~q13 을 원래대로 바꾼다. ⑪은 diffusion operator 에 해당하는 부분으로 3 vertex 에 해당하는 q0~q5 에 Hadamard gate 와 X gate 를 가한 후 q0~q4 를 control qubit 으로 하는 MCZ gate 를 가한다. Quantum oracle 과 마찬가지로 앞선 과정을 역순으로 진행하여 qubit 의 값들을 원래대로 바꾼다. 위와 같은 과정을 $\frac{\pi}{4}\sqrt{\frac{2^6}{6}}$회 반복한 뒤 measurement 를 진행하면 원하는 결과를 얻을 수 있다. Shots 을 1024 로 설정한 시뮬레이션 결과는 Figure 8 와 같다. 결과를 보면 찾고자 하는 state 들의 amplitude 가 다른 state 들의 amplitude 와 비교했을 때 큰 값을 가지는 것을 확인할 수 있지만 다른 state 들의 amplitude 도 적지 않게 나오는 것을 확인할 수 있다. 따라서 더 명확한 값을 얻기 위해 반복 횟수를 6 회로 늘려 시뮬레이션한 결과는 Figure 9 와 같다. 결과를 보면 $|000110\rangle, |001001\rangle, |010010\rangle, |011000\rangle, |100001\rangle, |100100\rangle$의 결과가 출력됨을 알 수 있다. 만약 $|00\rangle$을 빨강, $|01\rangle$을 초록, $|10\rangle$을 파랑이라 한다면 $|000110\rangle$의 결과는 첫번째

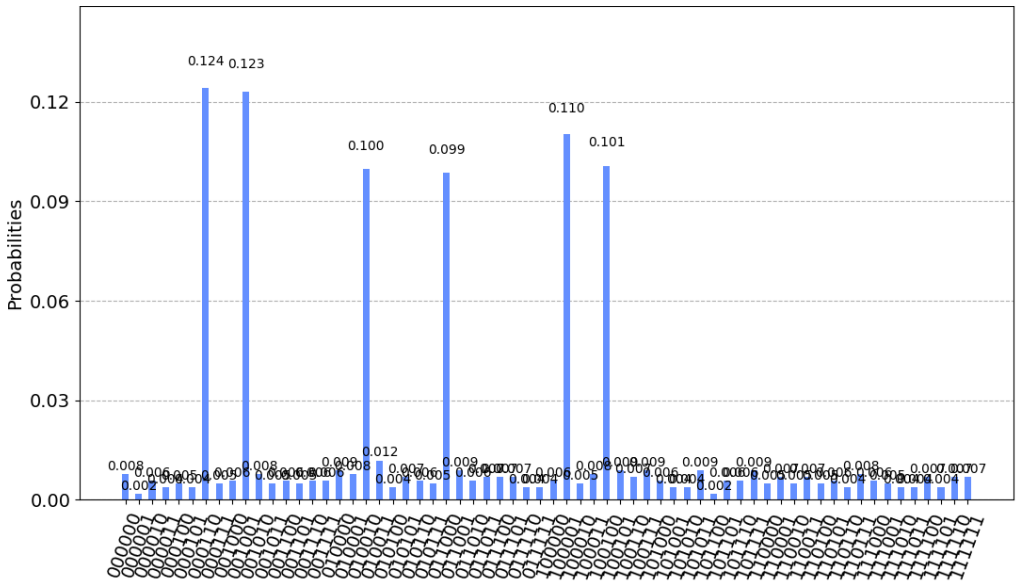

Figure 9. Graph 1 결과 (2회 반복)

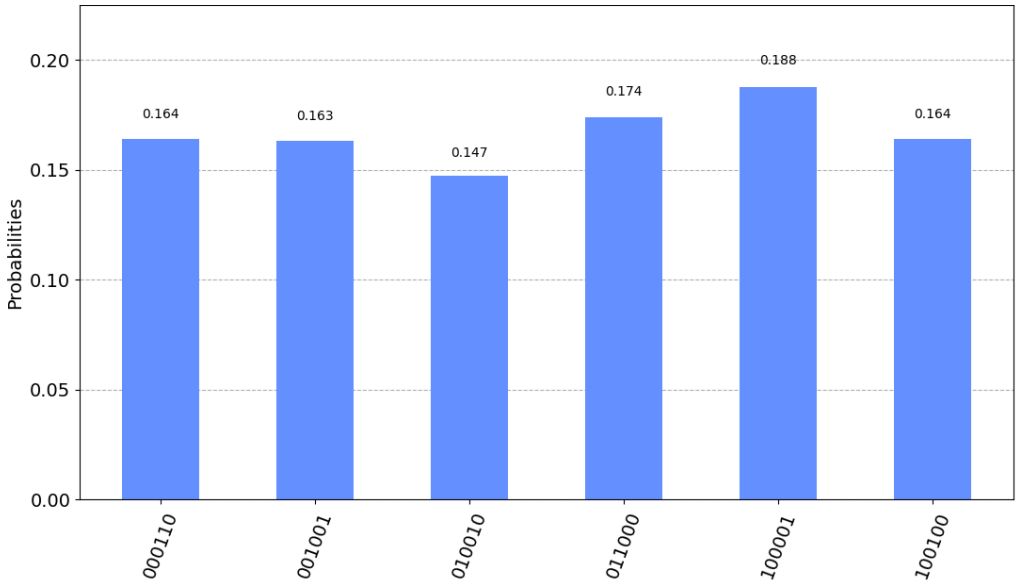

Figure 8. Graph 1 결과 (6회 반복)

vertex 가 파랑, 두번째 vertex 가 초록, 세번째 vertex 가 빨강을 가지는 state 임을 의미한다. 그 결과를 Figure 10 에 나타냈다. 따라서 회로가 잘 작동했음을 알 수 있다.

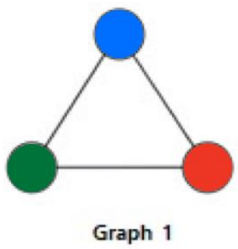

Figure 10. |000110⟩가 의미하는 graph

2. Graph 2

Graph 2 의 회로도 앞선 회로와 비슷하게 구성되고 Figure11 과 같다. 다만 Graph 2 에는 4 color 가 필요하므로 앞선 회로에서 color 11 을 가질 수 없도록 구성한 부분을 제거했다. ①에서는 initialization, ②~⑭는 quantum oracle, ⑮는 diffusion operator 을 구성한다. 앞선 Graph 1 의 회로와 비교했을 때 vertex 와 edge 의 수가 증가했으므로 그에 따라 color 를 나타내는 qubit 의 수도 증가하고 oracle 회로의 길이도 증가한 것을 확인할 수 있다. 해당 회로와 shots 를 1024 로 설정한 시뮬레이션 결과는 Figure 12 와 같다. 앞선 Graph 1 에서와 마찬가지로 더 명확한 답을 얻기 위해 반복횟수를 16 회로 늘려 시뮬레이션한 결과는 다음과 같다. 결과를 보면 각 vertex 의 color 가 서로 다른 24 개의 state 들이 출력됨을 알 수 있다. |00⟩을 빨강, |01⟩을 초록, |10⟩을 파랑, |11⟩을 보라라고 한다면 |00011011⟩의 결과는 첫번째 vertex 가 보라, 두번째 vertex 가 파랑, 세번째 vertex 가 초록, 네번째 vertex 가 빨강을 가지는 state 임을 의미한다. 그 결과를 Figure 14 에 나타냈다. 따라서 회로가 잘 작동했음을 알 수 있다.

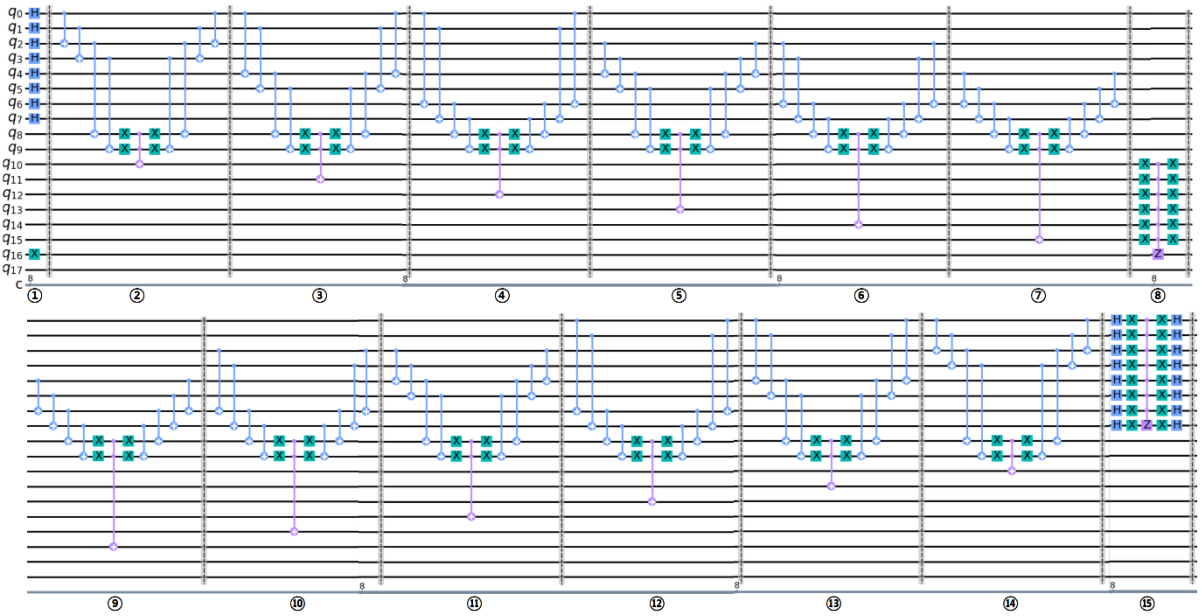

Figure 11. Graph 2 회로

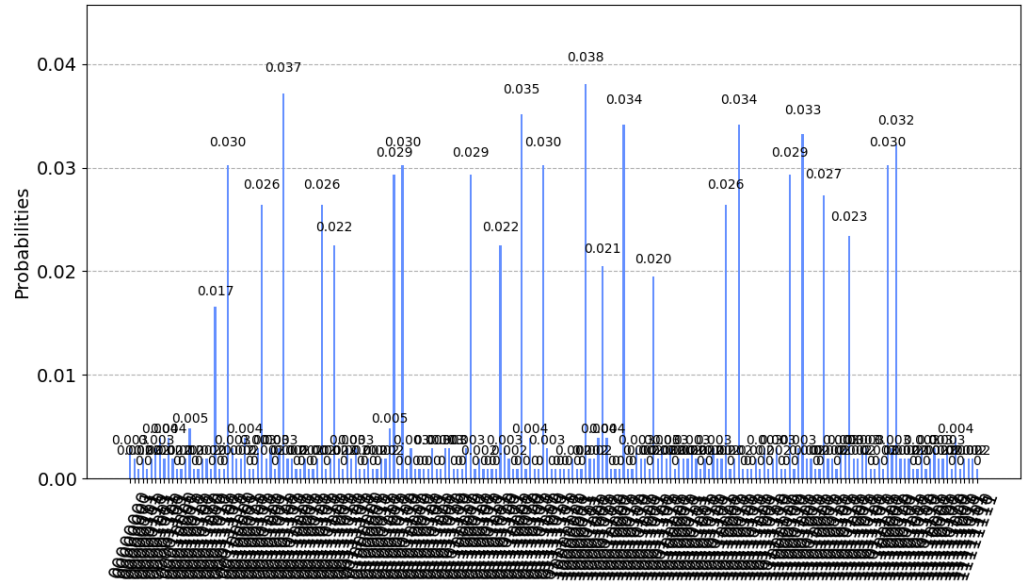

Figure 12. Graph 2 결과 (2회 반복)

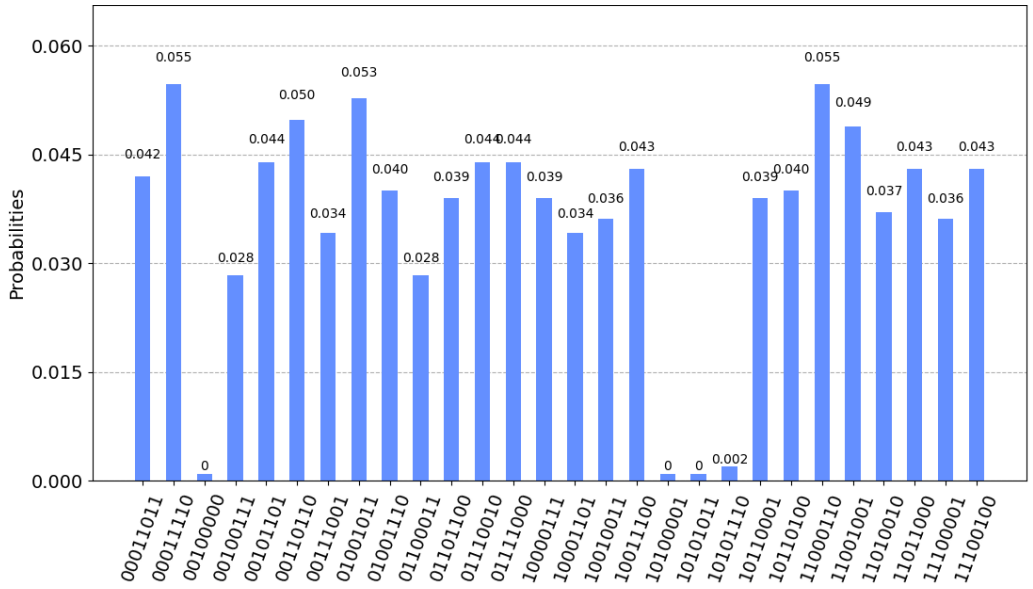

Figure 13. Graph 2 결과 (16회 반복)

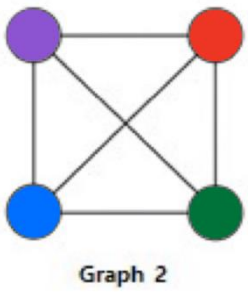

Figure 14. |00011011⟩가 의미하는 graph

다만 Table 2 에서는 Graph 1 의 회로에서 color 11 을 제외시켰던 부분과 같이 결과 값이 많이 나오는 것을 방지하기 위한 회로의 gate 는 생략했다. 불필요한 color 를 제거하는 부분은 vertex 와 edge 의 수에 따라 다르고 게다가 각 edge 가 vertex 에 어떻게 연결되어 있는지에 따라 다르기 때문이다.

C. 복잡도 계산

앞선 두 Graph 회로에서 gate 의 개수를 토대로 n vertex & m edge & k color graph coloring problem 을 해결하기 위한 회로의 gate 수를 예상해보면 Table1 과 같다. 이때 찾고자 하는 state 의 개수를 t 로 두었다. 그리고 Table 1 에 앞서 다뤘던 Graph 1, 2 를 대입하면 Table 2 와 같다.

		Initialization	Oracle	Diffusion	Total
n vertex m edge k color	H	$\lceil \log_2 k \rceil \times n$		$2 \times \lceil \log_2 k \rceil \times n$	$\lceil \log_2 k \rceil \times n + \left\lfloor \frac{\pi}{4} \sqrt{\frac{2^{\lceil \log_2 k \rceil \times n}}{t}} \right\rfloor \{2 \times \lceil \log_2 k \rceil \times n\}$
	X	1	$m \times \lceil \log_2 k \rceil \times 4 + m \times 2$	$2 \times \lceil \log_2 k \rceil \times n$	$1 + \left\lfloor \frac{\pi}{4} \sqrt{\frac{2^{\lceil \log_2 k \rceil \times n}}{t}} \right\rfloor \{m \times \lceil \log_2 k \rceil \times 4 + m \times 2 + 2 \times \lceil \log_2 k \rceil \times n\}$
	CX		$m \times \lceil \log_2 k \rceil \times 8$		$\left\lfloor \frac{\pi}{4} \sqrt{\frac{2^{\lceil \log_2 k \rceil \times n}}{t}} \right\rfloor \times \{m \times \lceil \log_2 k \rceil \times 8\}$
	MCX		$m \times 2$		$\left\lfloor \frac{\pi}{4} \sqrt{\frac{2^{\lceil \log_2 k \rceil \times n}}{t}} \right\rfloor \times m \times 2$
	MCZ		1	1	$\left\lfloor \frac{\pi}{4} \sqrt{\frac{2^{\lceil \log_2 k \rceil \times n}}{t}} \right\rfloor \times (1+1)$

Table 1. n vertex & m edge & k color graph 회로 구현에 필요한 gate 수

		Initialization	Oracle	Diffusion	Total
Graph 1 (3 vertex 3 edge 3 color 6 solution)	H	2×3		$2 \times 2 \times 3$	$2 \times 3 + 2\{2 \times 2 \times 3\}$
	X	1	$3 \times 2 \times 4 + 3 \times 2$	$2 \times 2 \times 3$	$1 + 2\{3 \times 2 \times 4 + 3 \times 2 + 2 \times 2 \times 3\}$
	CX		$3 \times 2 \times 8$		$2 \times \{3 \times 2 \times 8\}$
	MCX		3×2		$2 \times 3 \times 2$
	MCZ		1	1	$2 \times (1+1)$
		Initialization	Oracle	Diffusion	Total
Graph 2 (4 vertex 6 edge 4 color 24 solution)	H	2×4		$2 \times 2 \times 4$	$2 \times 4 + 2\{2 \times 2 \times 4\}$
	X	1	$6 \times 2 \times 4 + 6 \times 2$	$2 \times 2 \times 4$	$1 + 2\{6 \times 2 \times 4 + 6 \times 2 + 2 \times 2 \times 4\}$
	CX		$6 \times 2 \times 8$		$2 \times \{6 \times 2 \times 8\}$
	MCX		6×2		$2 \times 6 \times 2$
	MCZ		1	1	$2 \times (1+1)$

Table 2. Graph 1, 2의 gate 수

Ⅲ. 결론

본 논문에서는 graph coloring problem 을 Grover's algorithm 을 이용하여 해결하는 회로를 구현하였고 이를 Qiskit 을 이용하여 시뮬레이션을 실행해 $O(\sqrt{N})$의 복잡도로 graph coloring problem 이 해결 가능함을 확인하였다. 또한 이를 통해 n vertex & m edge & k color graph 를 해결하기 위한 회로에 필요한 gate 의 수를 예측하였다. 다만 qubit 은 한정된 자원이기 때문에 회로에서 color 을 나타내는 qubit 의 개수보다 중간 결과를 저장하기 위한 qubit 의 개수가 같거나 크다는 부분은 회로가 개선이 필요하다는 것을 의미한다. 개선이 된다면 같은 qubit 으로 더 많은 vertex 와 edge 를 가지는 graph coloring problem 을 해결할 수 있을 것이라 예상된다.

ACKNOWLEDGEMENT

이 성과는 2021 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2019R1A2C2010061)

참고문헌

[1] Michael A. Nielsen and Isaac L. Chuang, "Quantum Computation and Quantum Information", Cambridge University Press, 2010

[2] M. Boyer, G. Brassard, P. Høyer and A. Tapp, "Tight bounds on quantum searching", arXiv:quant-ph/9605034, May. 1996

[3] A. Younes, "Strength and Weakness in Grover's Quantum Search Algorithm", arXiv:0811.4481, Nov. 2008

[4] Lov K. Grover, "A fast quantum mechanical algorithm for database search", arXiv:quant-ph/9605043, Nov. 1996

[5] A. Saha, D. Saha and A. Chakrabarti, "Circuit Design for K-coloring Problem and it's Implement on Near-term Quantum Devices", arXiv:2009.06073, Sep. 2020

[6] A. Saha, A. Chongder, S. B. Mandal and A. Chakrabarti, "Synthesis of Vertex Coloring Problem Using Grover's Algorithm," 2015 IEEE International Symposium on Nanoelectronic and Information Systems, 2015, pp. 101-106, doi: 10.1109/iNIS.2015.55.