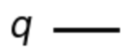
Quantum Challenge 2023 1 주차 보고서

서울대학교 전기정보공학부 이 상 연

1. Fair Quantum Coin Operator & Four-Sided Fair Quantum Coin Operator

1) Quantum Coin State

Quantum Coin state $|s\rangle \in H_c = \{a_0|0\rangle + a_1|1\rangle : a_0, a_1 \in \mathbb{C}\}$



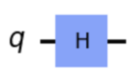
1 qubit register 의 up and down state 로 quantum coin state 표현한다.

2) Fair Quantum Coin Operator

Fair Coin Operator

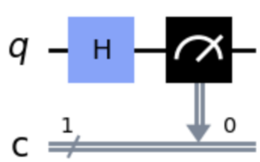
$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}\langle 0| + \frac{|0\rangle - |1\rangle}{\sqrt{2}}\langle 1|$$

Fair Quantum Coin Operator

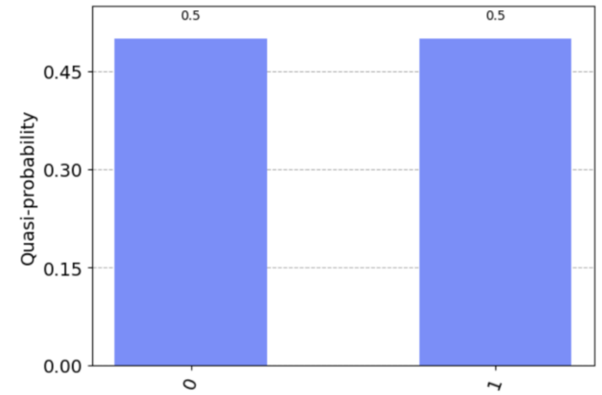


Hadamard gate 를 이용한다. Hadamard gate 는 0 state, 1 state 를 superposition state 로 만드는 역할이다. 각 superposition state 에서 0 state, 1 state 를 측정할 확률은 모두 0.5 이다.

Verifying Fair Quantum Coin Operator



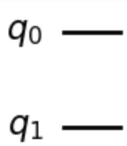
```
{'0': 4997, '1': 5003}
{'0': 0.5, '1': 0.5}
```



Fair quantum coin operator 를 1 회 적용한 후 0, 1 state 의 quasi-probability 를 측정했다. 각각 0.5 씩 측정되었다.

3) Four-Sided Quantum Coin State

Four sided Quantum Coin State $|s\rangle \in H_c' = \{a_0|0\rangle + a_1|1\rangle + a_2|2\rangle + a_3|3\rangle : a_0, a_1, a_2, a_3 \in \mathbb{C}\}$



0, 1, 2, 3 state 를 활용하여 four-sided quantum coin state 를 표현할 수 있으며, 0, 1, 2, 3 state 는 2 qubit register 로 encode 할 수 있다.

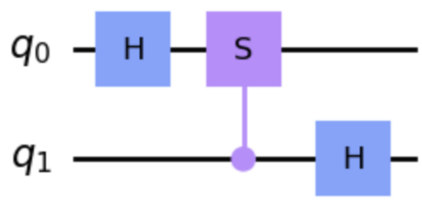
4) Four-Sided Fair Quantum Coin Operator

Four-Sided Fair Quantum Coin Operator

Four Sided Fair Coin Operator

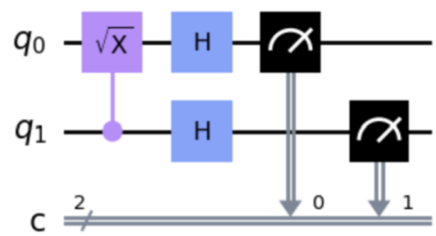
$$H' = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} = \frac{|0\rangle + |1\rangle + |2\rangle + |3\rangle}{\sqrt{4}}\langle 0| + \frac{|0\rangle + i|1\rangle - |2\rangle - i|3\rangle}{\sqrt{4}}\langle 1| + \frac{|0\rangle - |1\rangle + |2\rangle - |3\rangle}{\sqrt{4}}\langle 2| + \frac{|0\rangle - i|1\rangle - |2\rangle + i|3\rangle}{\sqrt{4}}\langle 3|$$

(Fourier Coin)

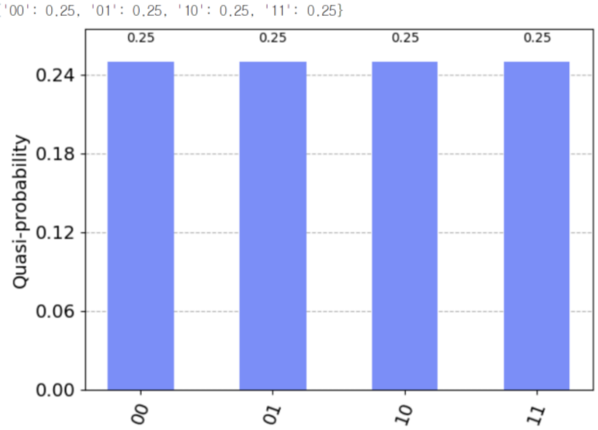


QFT 를 이용한다. N=4 일 때의 QFT matrix 를 이용한다. 0, 1, 2 ,3 state 를 superposition state 로 만드는 역할이다. 각 superposition state 에서 0, 1, 2 ,3 state 를 측정할 확률은 모두 0.25 이다.

Verifying Four-Sided Fair Quantum Coin Operator



{'00': 2486, '01': 2461, '10': 2558, '11': 2495}
{'00': 0.25, '01': 0.25, '10': 0.25, '11': 0.25}



Fair quantum coin operator 를 1 회 적용한 후 0, 1, 2, 3 state 의 quasi-probability 가 각각 0.25 씩 측정되었다.

5) Fairness Check

$|H_{ij}| = |\langle i|H|j\rangle| = \frac{1}{\sqrt{2}}$, all matrix elements of H has same absolute value $\frac{1}{\sqrt{N}}$,

$|H'_{ij}| = |\langle i|H'|j\rangle| = \frac{1}{\sqrt{4}}$ thus unbiased coin.

각 coin operator 의 entry 의 크기가 $\frac{1}{\sqrt{N}}$이므로 unbiased 라고 생각할 수 있다. 그리고 각 coin operator 의 verifying 작업에서도 unbiased 임을 확인했다.

2. Location on the Board

location on the board

$$|\psi\rangle \in H_p = \Big\{ \underbrace{\sum_{k=0}^{15} a_k |k\rangle}_{16\text{개 state}} : a_0, \cdots, a_{15} \in \mathbb{C} \Big\}$$

encoding by $|j_3 j_2 j_1 j_0\rangle$

$$|0000\rangle := |0\rangle$$
$$|0001\rangle := |1\rangle$$
$$\vdots$$
$$|1111\rangle := |15\rangle$$

총 16 칸 보드 각각을 state 관점에서 encode 하면 4 qubit register 가 필요하다. Start 지점을 0000 로, End 지점을 1111 로 encode 했다.

3. Quantum Coin with Shift Operator

1) Shift Operator

Right Shift Operator

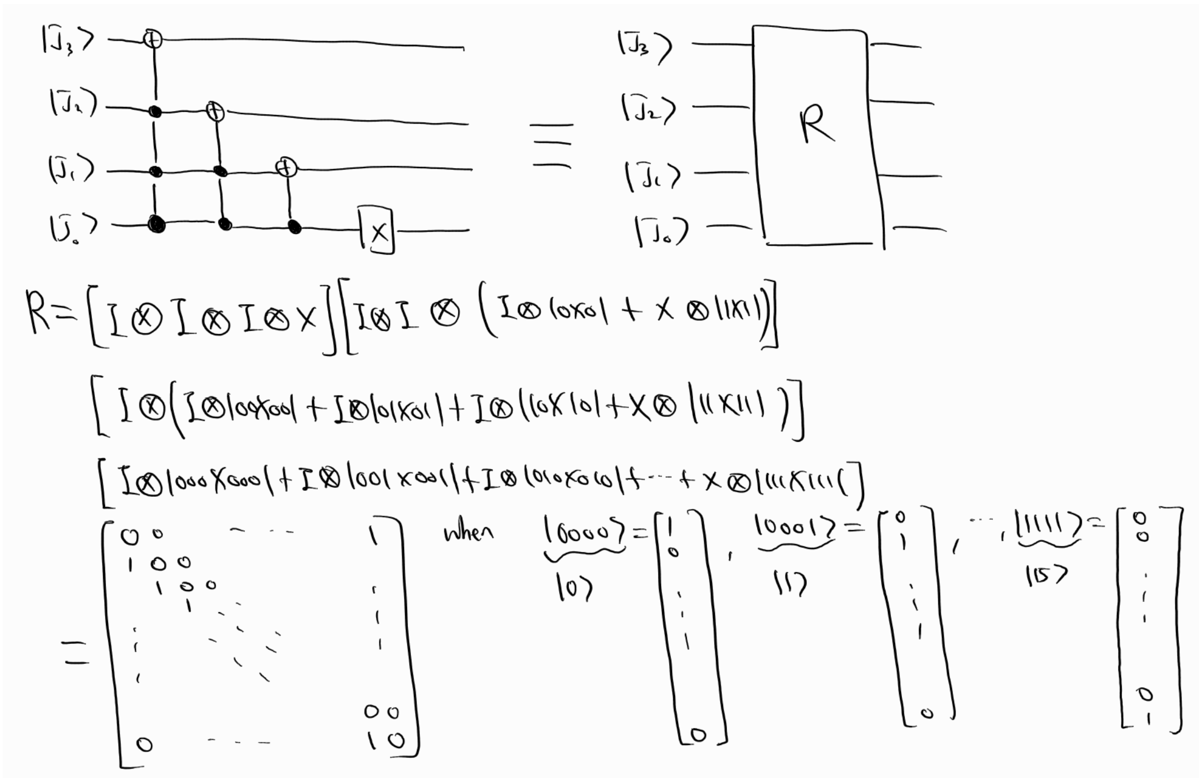
$$R = \begin{bmatrix} 0 & 0 & \cdots & & 1 \\ 1 & 0 & 0 & & 0 \\ \vdots & 1 & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \qquad L = \begin{bmatrix} 0 & 1 & \cdots & & 0 & 0 \\ 0 & 0 & 1 & & & \vdots \\ \vdots & & 0 & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \\ 0 & & & & & 1 \\ 1 & & \cdots & & & 0 \end{bmatrix}$$

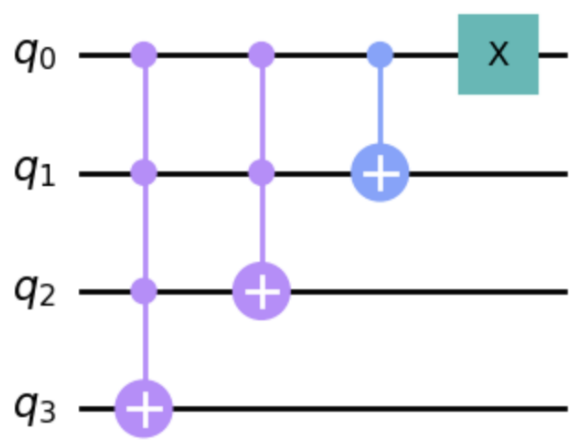
$$L = |15\rangle\langle 0| + \sum_{k=0}^{14} |k\rangle\langle k+1|$$
$$= |1111\rangle\langle 0000| + |0000\rangle\langle 0001| + |0001\rangle\langle 0010| + \cdots + |1110\rangle\langle 1111|$$
$$R = |0\rangle\langle 15| + \sum_{k=0}^{14} |k+1\rangle\langle k|$$
$$= |0000\rangle\langle 1111| + |0001\rangle\langle 0000| + |0010\rangle\langle 0001| + \cdots + |1111\rangle\langle 1110|$$

먼저 참고 논문과는 encoding 하는 방식이 달라 left shift operator 와 right shift operator 가 서로 뒤바뀌었음을 밝힌다. Operator 의 matrix representation 이 옳음을 확인한 operator 이다.

$|0000\rangle \coloneqq \begin{matrix} 1 \\ \vdots \\ 0 \end{matrix}$, …, $|1111\rangle \coloneqq \begin{matrix} 0 \\ \vdots \\ 1 \end{matrix}$로 encode 했다. 그리고 left operator, right operator 모두 unitary operator 를 만족해야 한다. 따라서 Right shift operator 는 15 state 를 0 state 로, left shift operator 는 0 state 를 15 state 로 shift 해야만 한다.

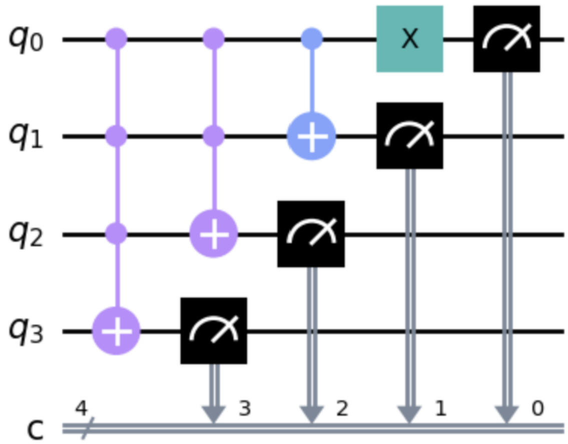
해당 operator 는 C3X operator, CCX operator, CNOT operator, X operator 로 구성할 수 있으며, tensor product 계산 과정은 아래 사진에 나타냈다.





Quantum Circuit 으로 나타내면 위 그림과 같다.

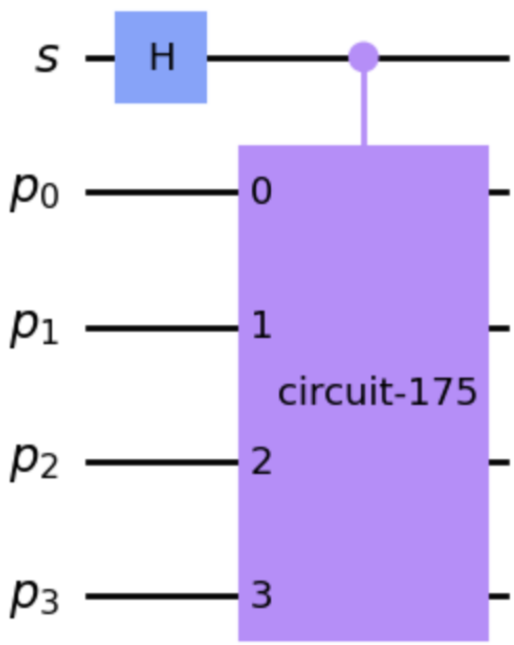
Verifying Right Shift Operator



여러 가지 initial state 에 right shift Operator 를 취한 결과, measurement 결과가 suitable 한 것을 확인했다.

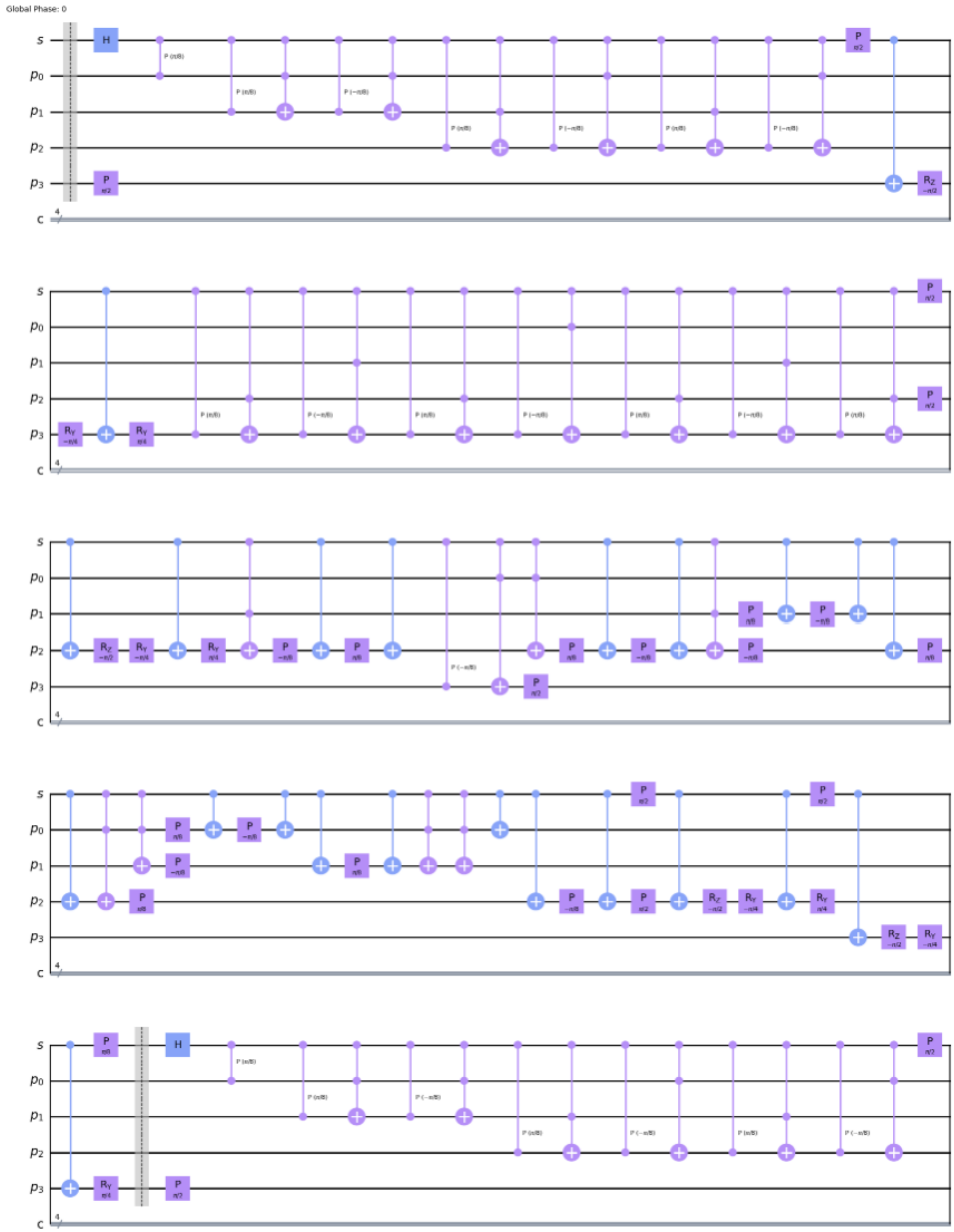
2) Shift Operator with Quantum Coin Operator

1 Step Board Evolution Operator



먼저 board 의 quantum system 은 Coin quantum register 와 position quantum register 의 composite state 로 구성 되어있다. 먼저 coin flip operator 를 적용한다. 그 후 coin state 가 0 일 때는 position register 에 identity matrix 를 취하고 coin state 가 1 일 때는 position register 에 shift operator 를 적용하는 것이 1 step evolution operator 이다.

N Step Board Evolution Operator



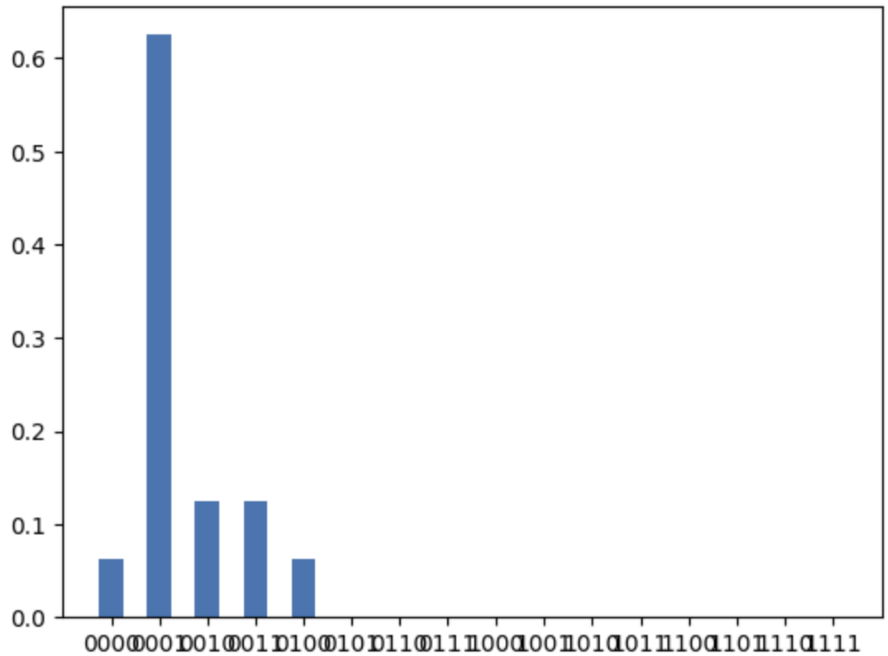
1 step board evolution operator 를 IONQ Cloud 에서 구현하기 위해서는 hardware 에서 구현할 수 있는 operator 로 decompose 하는 과정이 필요하다. Qiskit 의 trans file 함수를 활용하여 decompose 했다. 위

그림에서 barrier 와 barrier 사이가 1 개 step 이다. N 단계 구현을 위해서는 for 문을 활용하여 여러 번 circuit 을 generate 하는 과정을 거쳤다.

아래 결과는 N=4 일 때, 그리고 N=33 일 때의 각 state 가 측정될 확률을 shots=1000 으로 측정한 것을 막대 그래프로 시각화한 것이다. 소스코드는 N=0~49 까지 모든 자연수 N 에 대해 N 단계 소요 후 probability 를 측정하는 과정을 수행했다.

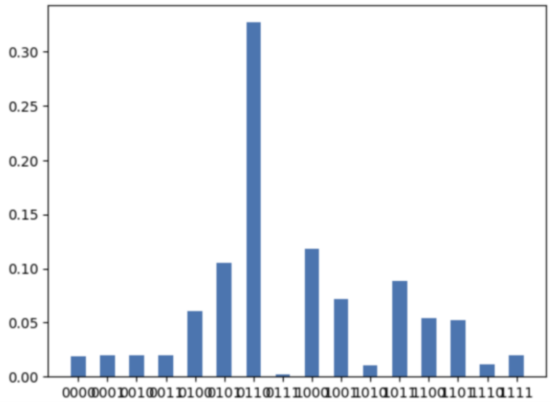
Verifying N Step Board Evolution Operator

```
4
{'0000': 48, '0001': 640, '0010': 130, '0011': 124, '0100': 58}
{'0000': 0.0625, '0001': 0.625, '0010': 0.125, '0011': 0.125, '0100': 0.0625}
```



N=4

33
{'0000': 26, '0001': 15, '0010': 14, '0011': 26, '0100': 64, '0101': 86, '0110': 358, '0111': 1, '1000': 106, '1001': 67, '1010': 6, '10
11': 89, '1100': 52, '1101': 49, '1110': 17, '1111': 24}
{'0000': 0.019192882, '0001': 0.019288786, '0010': 0.019597367, '0011': 0.019624859, '0100': 0.060641184, '0101': 0.10550928100000001,
'0110': 0.326655045, '0111': 0.002319723, '1000': 0.118175931, '1001': 0.071969516, '1010': 0.010329559, '1011': 0.088611752, '1100': 0.
054021015, '1101': 0.052439212, '1110': 0.011716500000000001, '1111': 0.019907385}



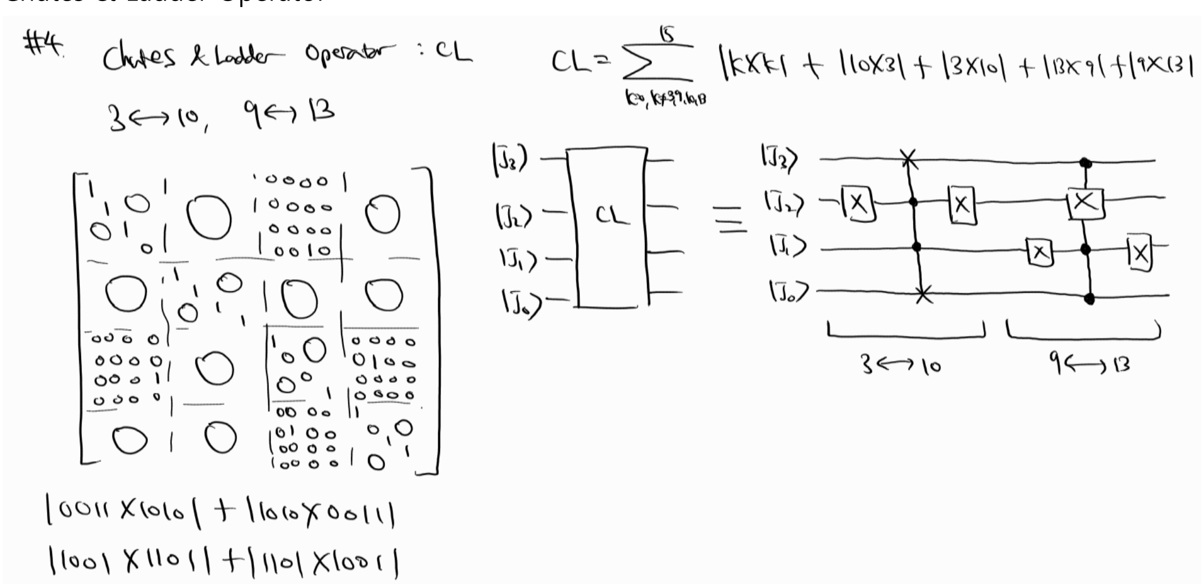
N=33

N 단계를 거치면서 quantum state 의 superposition 이 일어남을 확인할 수 있다. Classical Walk 에서는 점점 분포가 Gaussian 에 가까워지는 것과 달리 Quantum Walk 에서는 N 이 커지더라도 수렴하는 분포가 존재하지 않는다. 그리고 대칭적으로 분포가 생성되지 않는 것을 확인할 수 있는데, 이는 coin quantum operator 가 asymmetric 하기 때문이다.

4. Adding Chutes and Ladders in Board

1) Chutes & Ladder Operator (Notated CL)

Chutes & Ladder Operator



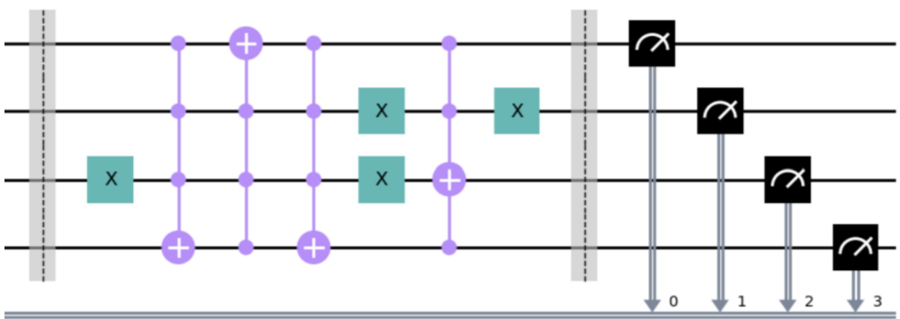
위의 그림과 같이 생각한 이유는

3 <-> 10 은 $|j_2 j_1\rangle = |01\rangle$일 때는 $|j_0\rangle$와 $|j_3\rangle$를 swap 하고, 나머지 경우에서는 identity matrix 를 취하는 것이기 때문에 controlled-swap gate 를 구성했다.

9 <-> 13 은 $|j_3 j_1 j_0\rangle = |101\rangle$일 때는 $|j_2\rangle$에 not gate 를 취하고, 나머지 경우에서는 identity matrix 를 취하는 것이기 때문에 C3X gate 를 구성했다. 3 <-> 10 와 9 <-> 13 를 직렬적으로 연결하면 chutes & ladder operator 가 완성된다.

따라서 전체 operator 는 coin state 에 coin quantum flip operator 를 가한 후, controlled – unitary operator 를 position register 에 가한다. 마지막으로 chutes & ladder operator 를 position register 에 취하는 것이 한 단계 evolution 하는 것이다.

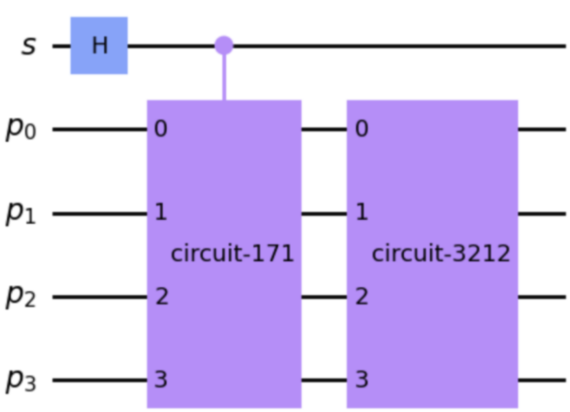
여러 가지 initial state 에 대해 Chutes & Ladder Operator 가 suitable 하게 작동하는지 파악하는 과정을 거쳤다. 결과 3 <-> 10, 9 <-> 13 순간 이동하는 것까지 정상적으로 작동하는 것을 확인했다.



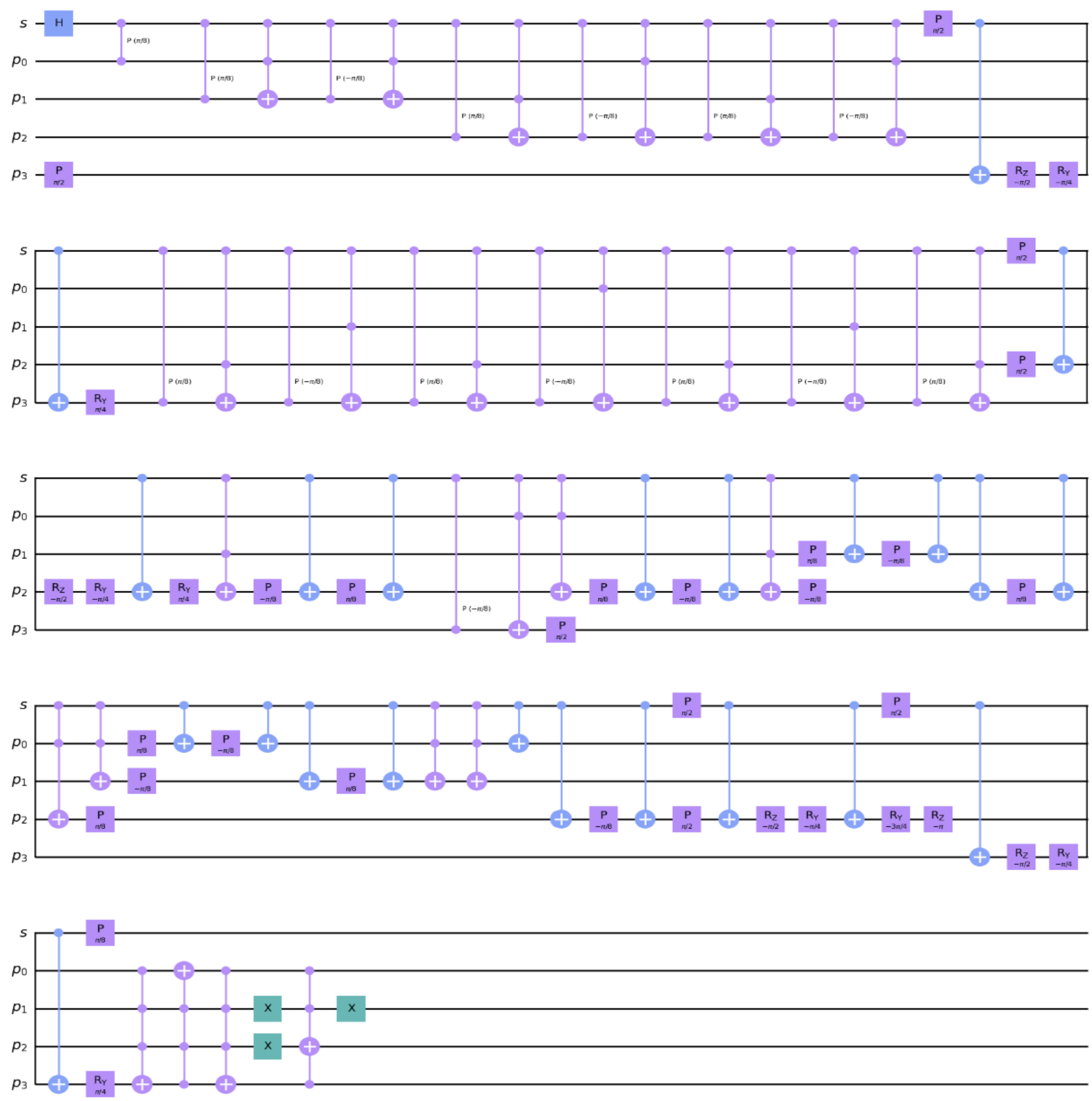
Chutes & Ladder Operator 를 hardware 에 맞게 decompose 한 circuit 이다. Controlled-Swap gate 는 CNOT gate 로 decompose 한 것을 확인할 수 있다.

2) Chutes & Ladders Board Evolution Operator

1 Step Chutes & Ladders Board Evolution Operator



전체 operator 를 hardware 에 맞게 decompose 하기 전 circuit 이다. 먼저 Coin State 에 Hadamard gate 를 취한 후, coin state 를 control bit 로 하여 controlled – shift operator 를 취한다. 다음으로 position state 에 1)에서 검증한 Chutes & Ladder operator 를 가한다. 각 operator 는 모두 unitary operator 이므로, 1 step operator 도 unitary operator 이다. 1 step 에 해당하는 operator 를 N step 시행하면 N 단계 이후의 superposition state 를 얻을 수 있다.



전체 1 board evolution operator (with Chutes & Ladder)를 IONQ hardware 에 맞게 decompose 한 circuit 이다. Controlled-shift operator 때문에 circuit 의 길이가 길어졌다.

5. Discussion

1) Role of Measurement in Quantum Chutes and Ladders Game

State measurement makes wave function collapse. Collapse of wave function arises randomness of Quantum Chutes and Ladders Game.

2) Measurement between Turns

Repetition of measurement and initializing states 는 wave function 을 collapse 시킨다. 그리고 turn 과 turn 사이에 measurement 를 시행한 후 다시 initialization 을 하는 것을 반복하는 것은 just repetition of 1 step evolution. Repetition of measurement can obtain probability of each state after 1 step evolution.

3) No Measurement between Turns

No measurement makes unitary evolution by each step. This procedure can obtain quantum superposition of states. 반복된 단계들이 superposition 을 일으킨다.

4) Quantum Analog of the "Memoryless" Nature of the Classical Game?

Quantum walks are quantum analogues of classical random walks.

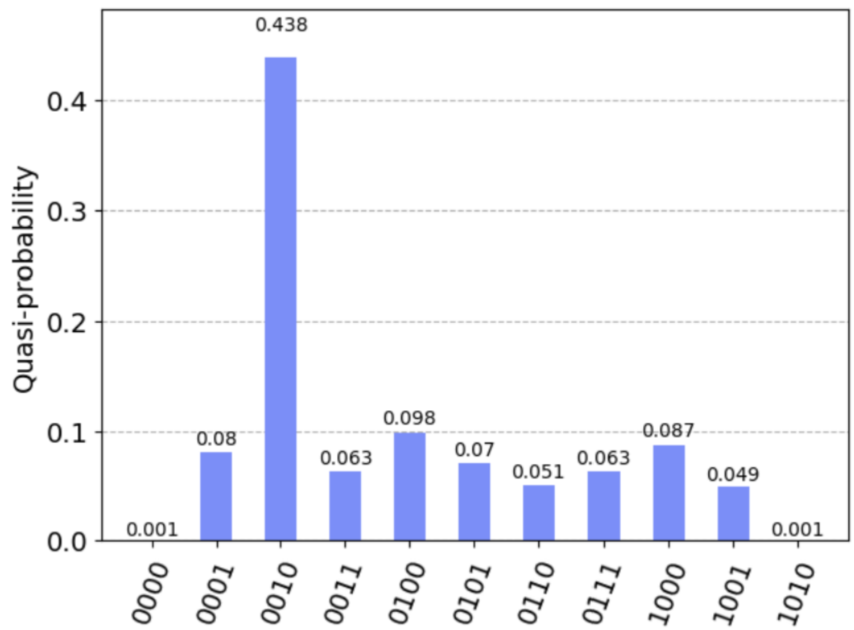
Classical Walk: Walker occupies definite states and randomness arises from stochastic transitions defined by Markov's Process.

Quantum Walk: Randomness arises through quantum superposition of states, non-random, reversible unitary evolution and collapse of the wave function due to state measurements.

6. Compare between #3 & #4 (10 Step)

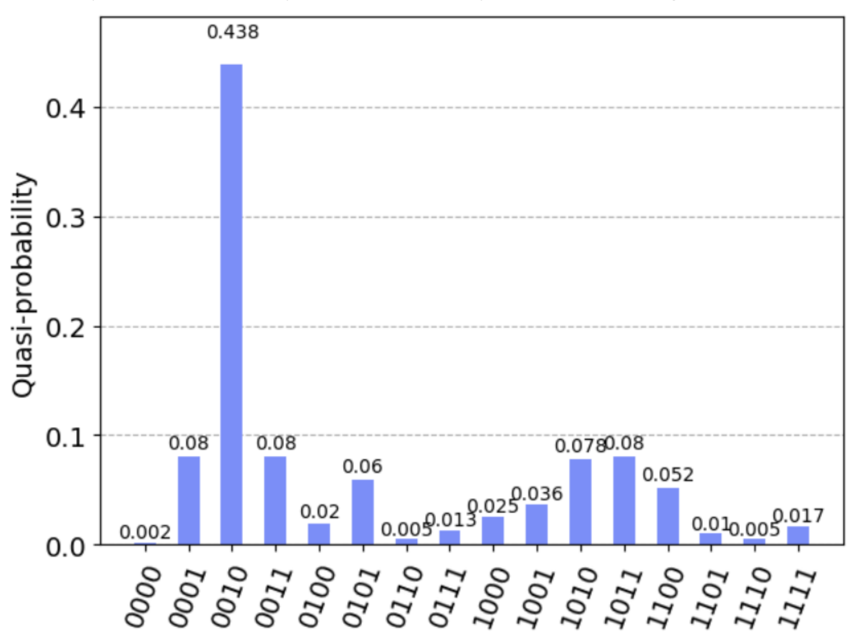
1) Task 3 (Not Containing Chutes & Ladders)

{'0000': 11, '0001': 787, '0010': 4341, '0011': 627, '0100': 1011, '0101': 726, '0110': 509, '0111': 621, '1000': 863, '1001': 49
5, '1010': 9}
{'0000': 0.000976562, '0001': 0.080078124, '0010': 0.438476562, '0011': 0.0625, '0100': 0.09765625, '0101': 0.0703125, '0110': 0.
05078125, '0111': 0.0625, '1000': 0.086914062, '1001': 0.048828124, '1010': 0.000976563}



2) Task 4 (Not Containing Chutes & Ladders)

{'0000': 18, '0001': 816, '0010': 4368, '0011': 783, '0100': 184, '0101': 589, '0110': 49, '0111': 119, '1000': 260, '1001': 366,
'1010': 759, '1011': 812, '1100': 566, '1101': 102, '1110': 44, '1111': 165}
{'0000': 0.001953125, '0001': 0.080078124, '0010': 0.438476562, '0011': 0.080078124, '0100': 0.01953125, '0101': 0.059570312, '01
10': 0.004882812, '0111': 0.012695312, '1000': 0.025390624, '1001': 0.036132813, '1010': 0.078125, '1011': 0.080078124, '1100':
0.051757812, '1101': 0.009765624, '1110': 0.004882812, '1111': 0.016601562}



우선 N=10 일 때 Not Containing 버전은 0000 ~ 1010 까지만 관측할 확률이 있다면, Containing 버전은 순간 이동하는 operator 로 인해 0000 ~ 1111 까지 모든 state 가 다 관측될 확률이 존재한다.

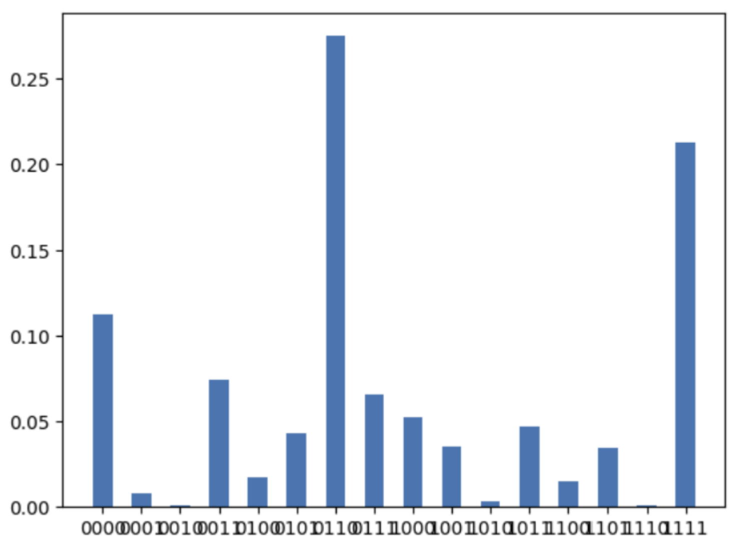
그리고 1000 state 의 경우. Containing 버전에서는 0011 과 1010 사이 ladder 때문에 0011 state 의 coefficient 가 1010 근방으로 많이 이동했다. 따라서 Containing 버전에서의 1000 state 의 coefficient 가 Not Containing 버전에서의 1000 state 의 coefficient 에 비해 작은 것을 확인할 수 있다.

그리고 1010 state 의 경우. Not Containing 버전에서는 1010 state 이 관측될 확률이 0.001 이지만, Containing 버전에서는 1010 state 이 관측될 확률이 0.078 이다. 이는 Chutes & Ladder 로 인해 지속적으로 0011 state 의 probability 가 이동했다고 해석할 수 있다. 그리고 9 <-> 13 ladder 이 존재하는 것도 마찬가지로 probability 의 trap 을 생성했다고 해석할 수 있다. 따라서 N 이 작을 때에는 1001 ~ 1101 사이에서 관측될 확률이 상대적으로 높음을 확인할 수 있다.

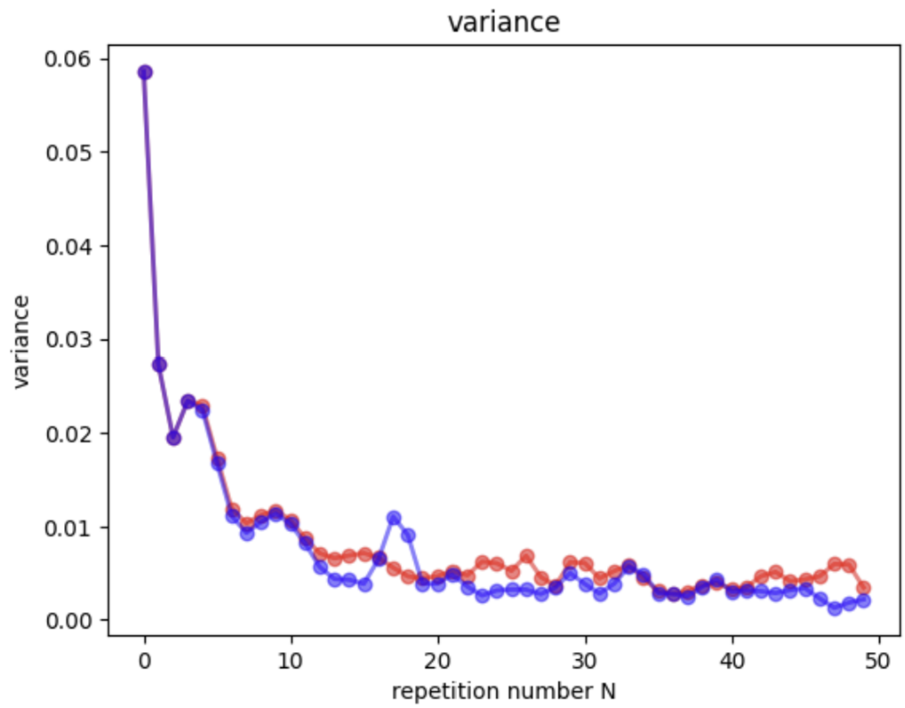
3) N Step Board Operator Containing Chutes & Ladders

Chutes & Ladders 를 포함한 버전에서도 N=0~49 까지 모든 자연수 N 에 대해 N 단계 소요 후 probability 를 측정하는 과정을 수행했다. 이 중 N=33 일 때 결과 그래프를 첨부하겠다.

33
{'0000': 146, '0001': 7, '0010': 1, '0011': 60, '0100': 14, '0101': 40, '0110': 267, '0111': 63, '1000': 70, '1001': 35, '1010': 3, '1011': 51, '1100': 10, '1101': 41, '1110': 0, '1111': 192}
{'0000': 0.112706712, '0001': 0.007829257999999999, '0010': 0.0009880380000000001, '0011': 0.074423598, '0100': 0.017590672, '0101': 0.043315038, '0110': 0.275062879, '0111': 0.06596548299999999, '1000': 0.05275086499999994, '1001': 0.035058029, '1010': 0.0037321940000000003, '1011': 0.047303868, '1100': 0.015321038, '1101': 0.034309361, '1110': 0.001023436, '1111': 0.212619532}



그러나 N 이 커지면 1001 ~ 1101 사이 Trap 이 사라지는 것을 확인할 수 있다. 그리고 N 에 따른 probability distribution 의 variance 그래프도 그렸다. 아래 그래프의 빨강색 선은 Chutes & Ladders 를 포함하지 않는 경우의 probability distribution 의 variance 그래프이다. 파란색 선은 Chutes & Ladders 를 포함하는 경우의 probability distribution 의 variance 그래프이다.



빨강색 그래프: Chutes & Ladders 를 포함하지 않는 경우의 probability distribution 의 variance 그래프
파란색 그래프: Chutes & Ladders 를 포함하는 경우의 probability distribution 의 variance 그래프

우선 classical walk 는 distribution 이 특정 distribution 으로 수렴하기 때문에 variance 역시 특정 값으로 수렴하는 것을 알 수 있는데, quantum walk 는 빨강색/파란색 그래프 모두 특정 값으로 수렴하지 않고 계속 fluctuation 이 있는 것을 확인했다. 이는 quantum walk 의 randomness 에 기인한다.

그리고 Chutes & Ladders 가 있을 때 variance 가 더 작은 것을 확인했다. Chutes & Ladders 가 다른 영역에서의 probability 를 교환하는 역할을 하기 때문에, probability 가 더 고르게 퍼지도록 만들기 때문이다. 따라서 파란색 그래프의 variance 가 빨강색 그래프의 variance 에 비해 작은 것을 확인했다.

그러나 Chutes & Ladders 가 있을 때, quantum walk 의 randomness 로 인해 N=16 ~ 19 사이에서 1000 state 에 probability 가 집중된다. (이는 소스코드 상에서 확인할 수 있다.) 따라서 N=16 ~ 19 사이에서 파란색 그래프의 variance 가 빨강색 그래프의 variance 에 비해 더 커지는 것을 알 수 있다.