# Supplementary Material for the Article: Implementation and Evaluation of a Technique to Construct Quantum Bayesian Networks

Karthick Seshadri[1*], Awismrit Parida[1], Srinath Devale[2], Padmil Nayak[1]

[1] Department of Computer Science and Engineering
National Institute of Technology, Andhra Pradesh
Tadepalligudem, 534101, Andhra Pradesh, India.

[2] Department of Artificial Engineering and Machine Learning
Ballari Institute of Technology and Management
Ballari, 583101, Karnataka, India.

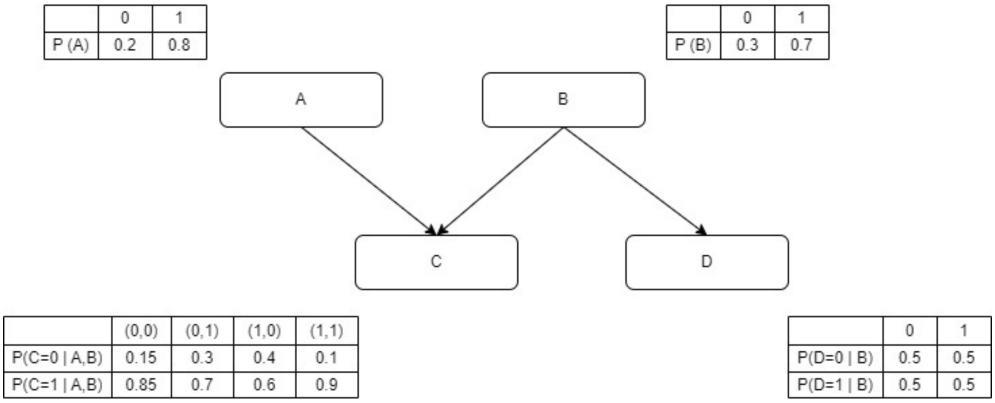
*Corresponding author(s). E-mail(s): karthick.seshadri@nitandhra.ac.in;
Contributing authors: wsmrt19@gmail.com; srinath.d@bitm.edu.in;
mtcs2205@student.nitandhra.ac.in;

## 1 Construction of quantum Bayesian network

In this section, a step-by-step implementation of the algorithm to convert a Bayesian network into a quantum circuit is illustrated with the help of an example Bayesian network shown in Figure 1.

Figures 2a and 2b show the quantum circuit after processing nodes A and B, respectively. A rotation transformation with an angle corresponding to their state probabilities is applied to individual qubits q0 and q1.

Figure 3 shows the state of the quantum circuit after incorporating the representation for node C. An ancilla qubit q0 can be seen added to the circuit as node C has two parent nodes. A node q3 corresponding to node C is added to the circuit with controlled operations from q0, q1 and q2. After the addition of qubit q4 corresponding to node D, the final circuit is shown in Figure 4. As node D is dependent only on node B, the qubit q4 has controlled transformations only from q2.



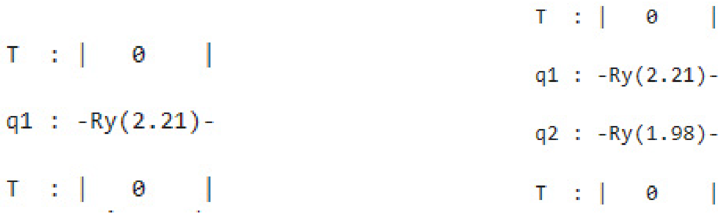
| | 0 | 1 |
|---|---|---|
| P (A) | 0.2 | 0.8 |

| | 0 | 1 |
|---|---|---|
| P (B) | 0.3 | 0.7 |

| | (0,0) | (0,1) | (1,0) | (1,1) |
|---|---|---|---|---|
| P(C=0 \| A,B) | 0.15 | 0.3 | 0.4 | 0.1 |
| P(C=1 \| A,B) | 0.85 | 0.7 | 0.6 | 0.9 |

| | 0 | 1 |
|---|---|---|
| P(D=0 \| B) | 0.5 | 0.5 |
| P(D=1 \| B) | 0.5 | 0.5 |

**Fig. 1**: An illustrative 4-node Bayesian network with conditional probability tables of its nodes.



```
T  : |   0    |

q1 : -Ry(2.21)-

T  : |   0    |
```

(a) State of the circuit after processing node A of the Bayesian network in Figure 1. (q1: A)

```
T  : |   0    |

q1 : -Ry(2.21)-

q2 : -Ry(1.98)-

T  : |   0    |
```

(b) State of the circuit after processing node B of the Bayesian network in Figure 1. (q1: A, q2: B)

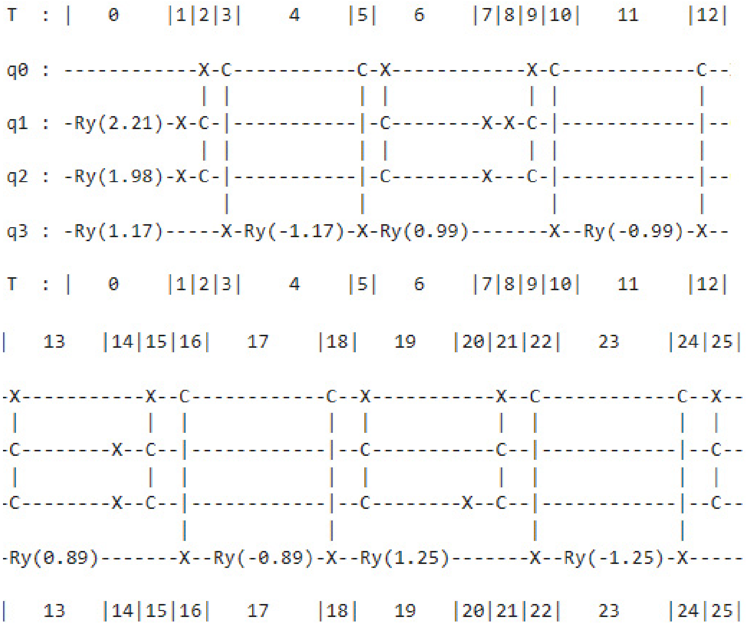
**Fig. 2**: Circuit after adding root nodes

## 2 Representation of Root Nodes

A step-by-step procedure to incorporate a single root node in the quantum circuit is discussed in Algorithm 2. The algorithm reserves initial $(i_d - 1)$ qubits as ancilla qubits and adds a qubit corresponding to the current root node into the circuit. The node probabilities are realized by applying the qubit rotation across the y-axis.

## 3 Representation of non-root Nodes

Incorporating a non-root node in the quantum circuit is relatively tricky compared to the root nodes, because of conditional dependence on one or more parent nodes. Algorithm 3 is responsible for adding non-root nodes to the quantum circuit during the construction process. It starts by assigning a qubit to the current node $(v)$ and then creating a list of parent nodes of $v$ and a list (cQbits) that maps the parent nodes to their corresponding qubits in the circuit. For each combination of the states of the parent nodes, the $C$-Ry gate or $C^n$Ry gate is applied to the target qubit, where



```
T  : |   0    |1|2|3|    4    |5|   6    |7|8|9|10|  11    |12|

q0 : ------------X-C-----------C-X------------X-C------------C--
                 | |           | |            | |            |
q1 : -Ry(2.21)-X-C-|-----------|-C--------X-X-C-|------------|--
                 | |           | |            | |            |
q2 : -Ry(1.98)-X-C-|-----------|-C--------X---C-|------------|--
                   |           |                |            |
q3 : -Ry(1.17)-----X-Ry(-1.17)-X-Ry(0.99)-------X--Ry(-0.99)-X--

T  : |   0    |1|2|3|    4    |5|   6    |7|8|9|10|  11    |12|

|   13   |14|15|16|   17    |18|   19   |20|21|22|   23    |24|25|

-X-----------X--C------------C--X-----------X--C------------C--X--
 |           |  |            |  |           |  |            |  |
-C--------X--C--|------------|--C-----------C--|------------|--C--
 |           |  |            |  |           |  |            |  |
-C--------X--C--|------------|--C--------X--C--|------------|--C--
                |            |                 |            |
-Ry(0.89)-------X--Ry(-0.89)-X--Ry(1.25)-------X--Ry(-1.25)-X-----

|   13   |14|15|16|   17    |18|   19   |20|21|22|   23    |24|25|
```

**Fig. 3**: State of the circuit after processing node C of the Bayesian network in Figure 1. (q0: ancilla, q1: A, q2: B and q3: C)

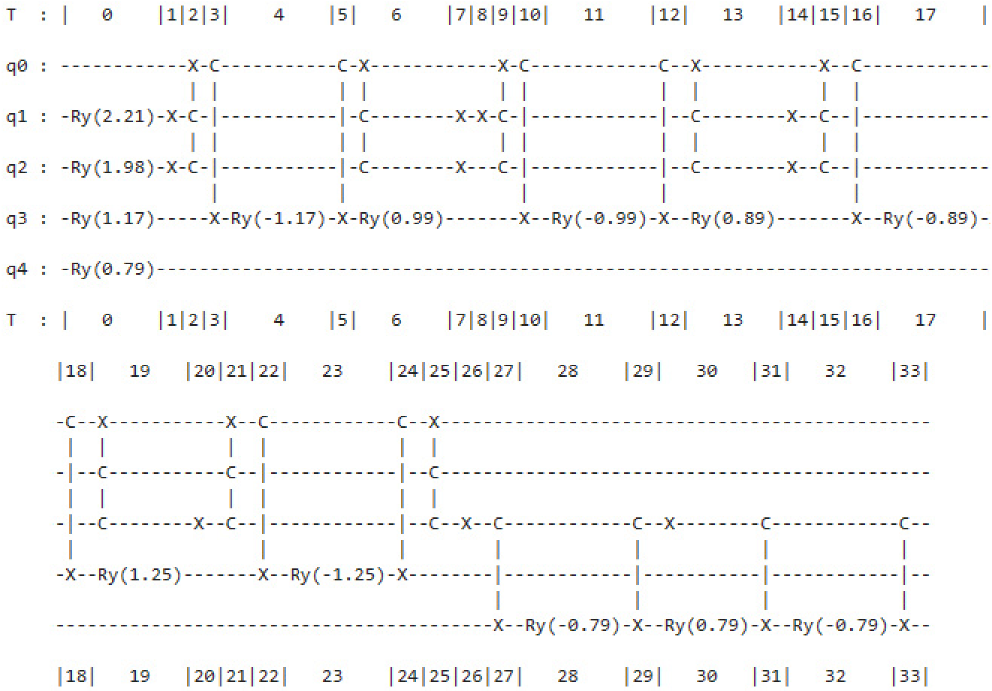
$n$ corresponds to the number of parents of $v$. Circuit realizations for the $C$-Ry and $C^n$Ry gates, necessary for the implementation are performed using CNOT, CCNOT, and Ry gates as described in the next subsections.

## 4 $C$-Ry Gate

A quantum circuit of two qubits that has a $C$-Ry gate with q0 as the control qubit and q1 as the target qubit is shown in Figure 5. The intuition behind applying the series of gates is explained in the main article. Algorithm 4 realizes the $C$-Ry transformation by utilizing repeated Ry and CNOT operations.

## 5 $C^n$-Ry Gate

Algorithm 5 illustrates the sequence of gates required to implement the $C^n - Ry$ gate on a quantum circuit. The circuit for the 4-qubit controlled rotation gate, $C^4$-Ry in Figure 6, consists of three parts and utilizes (n-1) ancilla qubits, all initialized and ending in the state $|0\rangle$. The first stage of the circuit aims to combine all the control bits q0, q1, q2, and q3 to obtain their product $q0 \cdot q1 \cdot q2 \cdot q3$. This is accomplished by applying CCNOT gates. The first CCNOT gate operates on q0 and q1, modifying the state of the first ancilla qubit to $|q0 \cdot q1\rangle$. Subsequently, each CCNOT gate acts on the next control qubit and the current product, updating the state of the corresponding



```
T  : |   0    |1|2|3|    4    |5|   6    |7|8|9|10|    11   |12|   13   |14|15|16|   17    |

q0 : ------------X-C-----------C-X------------X-C------------C--X-----------X--C------------
                 | |           | |            | |            |  |           |  |
q1 : -Ry(2.21)-X-C-|-----------|-C--------X-X-C-|------------|--C--------X--C--|------------
                 | |           | |            | |            |  |           |  |
q2 : -Ry(1.98)-X-C-|-----------|-C--------X---C-|------------|--C--------X--C--|------------
                   |           |                |            |                 |
q3 : -Ry(1.17)-----X-Ry(-1.17)-X-Ry(0.99)-------X--Ry(-0.99)-X--Ry(0.89)-------X--Ry(-0.89)-

q4 : -Ry(0.79)------------------------------------------------------------------------------

T  : |   0    |1|2|3|    4    |5|   6    |7|8|9|10|    11   |12|   13   |14|15|16|   17    |

     |18|   19   |20|21|22|   23    |24|25|26|27|   28    |29|   30   |31|   32    |33|

     -C--X-----------X--C------------C--X----------------------------------------------
      |  |           |  |            |  |
     -|--C-----------C--|------------|--C----------------------------------------------
      |  |           |  |            |  |
     -|--C--------X--C--|------------|--C--X--C------------C--X--------C------------C--
      |                 |            |        |            |           |            |
     -X--Ry(1.25)-------X--Ry(-1.25)-X--------|------------|-----------|------------|--
                                              |            |           |            |
     -----------------------------------------X--Ry(-0.79)-X--Ry(0.79)-X--Ry(-0.79)-X--

     |18|   19   |20|21|22|   23    |24|25|26|27|   28    |29|   30   |31|   32    |33|
```

**Fig. 4**: State of the circuit after processing node D of the Bayesian network in Figure 1. (q0: ancilla, q1: A, q2: B, q3: C and q4:D )

**Algorithm 2** AddRootNodes (*circuit*: Quantum circuit object, *B_N*: Bayesian Network object, $v$: Node being processed, *node_qbit_no*: Dictionary of nodes mapped to their circuit qubit number, $i_d$: Maximum in-degree of the Bayesian network)

```
    returns
    node_qbit_no: Updated dictionary of nodes mapped to their circuit qubit number

if dictionary node_qbit_no is empty then
    t_qbit = $i_d$-1
else
    t_qbit = Quantum circuit's last qubit index + 1
end if
$P(v = 1) \leftarrow B\_N.net[v]['probability']$
$\theta$ = probToAngle ($P(v = 1)$)
Apply the Ry gate to the target qubit with angle $\theta$ .
Update the node_qbit_no dictionary with the newly added qubit.
return node_qbit_no
```

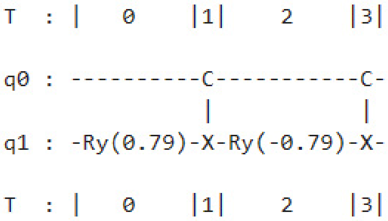
**Algorithm 3** AddNonRootNodes (*circuit*: Quantum circuit object, *B_N*: Bayesian Network object, *v*: Node being processed, *node_qbit_no*: Dictionary of nodes mapped to their circuit qubit number, $i_d$: Maximum in-degree of the Bayesian network)

```
    returns:
    node_qbit_no: Updated dictionary of nodes mapped to their circuit qubit number

parent[ ] ← List of parents of node v
cQbits[ ] ← $ConvertToList(\bigcup_{\text{node}\in\text{parent}} node\_qubit\_no[\text{node}])$
tQbit ← Maximum qubit index utilized in the circuit + 1

for each $c \in$ Combination of states of the parent nodes do
    Apply $X$ gate in circuit to qubits with 0 state for combination c.
    prob ← Retrieve $c$'s conditional probability from $B\_N.net[v]$
    if length(cQbits) == 1 then
        $C\_Ry$(circuit, prob, cQbits[0], tQbit)
    else
        $q_a$[ ] = [0, ..., $i_d$]
        $q_u$[ ] = List of occupied qubits of the circuit in sorted order.
        $C^n\_Ry$(circuit, prob, sorted(cQbits), $q_u$, $q_a$)
    end if
    Apply $X$ gate in circuit to qubits with 0 state for combination c.
end for

node_qbit_no[v] ← tQbit
return node_qbit_no
```



```
T  : |    0     |1|     2     |3|

q0 : ----------C-----------C-
               |           |
q1 : -Ry(0.79)-X-Ry(-0.79)-X-

T  : |    0     |1|     2     |3|
```

**Fig. 5**: Illustrative representation of C-Ry gate generated by Algorithm 5 with q0 as control qubit and q1 as target qubit.

ancilla qubit. This process repeats until the final ancilla qubit represents the state $|q0 \cdot q1 \cdot q2 \cdot q3\rangle$.

In the second stage, C-Ry gate is applied on the target qubit (q7), conditioned on the final ancilla qubit. In other words, the *Ry* gate is applied if and only if all of the control bits q1 through q3 are in the state $|1\rangle$.

**Algorithm 4** *C_Ry* ($P(|1\rangle)$: Probability, *circuit*: Quantum circuit object, cQubit: Control qubit, tQubit: Target qubit)

```
returns
    circuit: a modified quantum circuit.

$\theta$ = probToAngle ($P(|1\rangle)$)
    ▷ Convert the probability of ket 1 to an angle.
circuit.Ry($tQbit, \theta/2$)
    ▷ Apply a RY gate to the target qubit with angle $\theta/2$.
circuit.CNOT($cQbit, tQbit$)
    ▷ Apply a CNOT gate controlled by the cQbit and targeting the tQbit.
circuit.Ry($tQbit, -\theta/2$)
    ▷ Apply a RY gate to the target qubit with angle $-\theta/2$
circuit.CNOT($cQbit, tQbit$)
    ▷ Apply a CNOT gate controlled by the cQbit and targeting the tQbit.
return circuit
```

The final stage of the circuit reverses the steps taken in the first stage, returning all the ancilla qubits to their initial state $|0\rangle$. By combining these three stages, the circuit achieves the desired outcome of applying the $C^4$-Ry gate to the target qubit.

# 6 Time-complexity of Quantum Bayesian Network Construction

The time complexity of all the algorithms discussed here is mentioned in Table 1, in which $i_{dmax}$ refers to the maximum in-degree across nodes in the Bayesian Network and $C_{max}$ refers to the maximum number of entries in a conditional probability table across all nodes in the Bayesian network.

# 7 Convergence of Error and Extended Error Analysis

The results in Table 2 show that QJP errors converge rapidly with increasing shots, achieving accuracy comparable to classical inference with significantly fewer samples. Table 3 presents an extended error analysis for the Cancer network, showing that QJP consistently produces lower or comparable deviations relative to CJP, with maximum errors below 0.01. These findings underscore the efficiency and robustness of quantum Bayesian network inference.

**Algorithm 5** $C^n\_Ry$ (*circuit*: Quantum circuit object, $P(|1\rangle)$: Probability of ket 1, $q_c[\ ]$: List of control qubits, $q_u[\ ]$: List of already occupied qubits, $q_a[\ ]$: List of ancilla qubits)

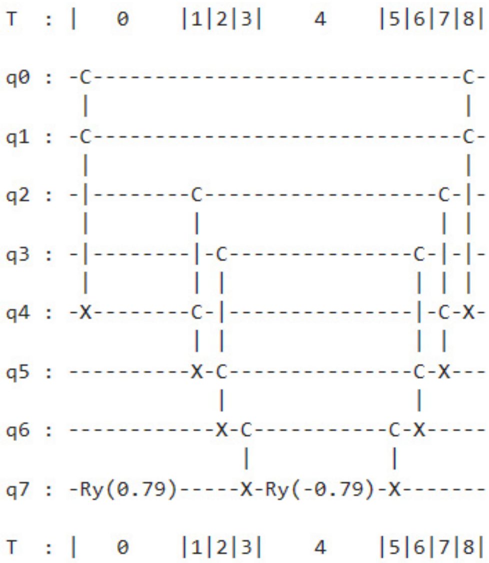
```
$n \leftarrow$ length of $q_c[\ ]$, $s \leftarrow q_c[0]$, $mem\_target \leftarrow$ empty list
circuit.CCNOT($q_c[0]$, $q_c[1]$, $q_a[0]$)
▷ Apply CCNOT gate to the circuit with qubit# $q_c[0]$, $q_c[1]$ as control qubits and $q_a[0]$ as the target qubit.
$prev\_target \leftarrow q_a[0]$

if $n > 2$ then
    for index=2 to length($q_c$) do
        i= $q_c$[index]
        circuit.CCNOT(i, prev_target, prev_target + 1)
        ▷ Apply CCNOT gate with qubit# $i$, $prev\_target$ as control qubits and $prev\_target + 1$ as the target qubit.
        Append [$i$, $prev\_target$, $prev\_target + 1$] to the mem_target list
        $prev\_target \leftarrow prev\_target + 1$
    end for
end if

$\theta$ = probToAngle ($P(|1\rangle)$)
circuit.Ry($q_u[last\_index] + 1$, $\theta/2$)
▷ Apply Ry gate with target qubit# $q_u[last\_index] + 1$ and angle $\theta/2$
circuit.CNOT($prev\_target$, $q_u[last\_index] + 1$)
▷ Apply CNOT gate with control qubit# $prev\_target$ and target qubit# $q_u[last\_index] + 1$
circuit.Ry($q_u[last\_index] + 1$, $-\theta/2$)
▷ Apply Ry gate with target qubit# $q_u[last\_index] + 1$ and angle $-\theta/2$
circuit.CNOT($prev\_target$, $q_u[last\_index] + 1$)

if $n > 2$ then
    for i= length(mem_target)-1 down to 0 do
        qPos = mem_target[i]
        $q\_1, q\_2, q\_3 \leftarrow qPos[0], qPos[1], qPos[2]$
        circuit.CCNOT($q\_1, q\_2, q\_3$)
        ▷ Apply CCNOT gate to circuit with qubit# q_1, q_2 as control qubits and q_3 as the target qubit.
    end for
end if

circuit.CCNOT($q_c[0]$, $q_c[1]$, $q_a[0]$)
▷ Apply CCNOT gate with qubit# $q_c[0]$, $q_c[1]$ as control qubits and $q_a[0]$ as the target qubit.
return circuit
```



```
T  : |   0    |1|2|3|    4    |5|6|7|8|

q0 : -C------------------------------C-
      |                              |
q1 : -C------------------------------C-
      |                              |
q2 : -|--------C-------------------C-|-
      |        |                   | |
q3 : -|--------|-C---------------C-|-|-
      |        | |               | | |
q4 : -X--------C-|---------------|-C-X-
               | |               | |
q5 : ----------X-C---------------C-X---
                 |               |
q6 : ------------X-C-----------C-X-----
                   |           |
q7 : -Ry(0.79)-----X-Ry(-0.79)-X-------

T  : |   0    |1|2|3|    4    |5|6|7|8|
```

**Fig. 6**: Illustrative representation of $C^4$-Ry gate generated by Algorithm 5 with q0, q1, q2, q3 as control qubits and q7 as the target qubit. (q4, q5, q6 are work qubits/ ancilla qubits)

**Table 1**: Time complexities

| Algorithm | Time Complexity |
|---|---|
| $C - Ry$ | $O(1)$ |
| $C^n - Ry$ | $O(\|q_c\|)$ |
| AddRootNodes | $O(1)$ |
| AddNonRootNodes | $O(C_{max}.i_{dmax})$ |
| Bn_To_Circuit | $O(\|V\|.C_{max}.i_{dmax})$ |
| $C^n - Z$ | $O(\|q_c\|)$ |

**Table 2**: Convergence of QJP and CJP errors as a function of sample/shot size in Cancer, Earthquake, and Asia networks

| Network | Shots (Quantum) | Error-QJP | Samples (Classical) | Error-CJP |
|---|---|---|---|---|
| Cancer | 100 | 0.0052 | 1000 | 0.0185 |
| | 500 | 0.0021 | 5000 | 0.0123 |
| | 1000 | 0.0011 | 10000 | 0.0110 |
| Earthquake | 100 | 0.0048 | 1000 | 0.0021 |
| | 500 | 0.0019 | 5000 | 0.0009 |
| | 1000 | 0.0006 | 10000 | 0.0005 |
| Asia | 100 | 0.0061 | 1000 | 0.0009 |
| | 500 | 0.0027 | 5000 | 0.0003 |
| | 1000 | 0.0005 | 10000 | 0.0001 |

**Table 3**: Extended error table: Per-state evaluation of QJP and CJP against ground truth in the Cancer Bayesian Network

| Quantum State | QJP | CJP | Ground Truth | Error-QJP | Error-CJP |
|---|---|---|---|---|---|
| 00000 | 0.3524 | 0.3596 | 0.3556 | 0.0032 | 0.0040 |
| 00001 | 0.0881 | 0.1480 | 0.0847 | 0.0034 | 0.0633 |
| 00010 | 0.1510 | 0.0870 | 0.1508 | 0.0002 | 0.0638 |
| 00011 | 0.0378 | 0.0372 | 0.0357 | 0.0021 | 0.0015 |
| 00100 | 0 | 0 | 0.0002 | 0.0002 | 0.0002 |
| 00101 | 0.0002 | 0 | 0.0002 | 0 | 0.0002 |
| 00110 | 0 | 0.0001 | 0 | 0 | 0.0001 |
| 00111 | 0.0004 | 0.0005 | 0.0003 | 0.0001 | 0.0002 |
| 01000 | 0.1467 | 0.1434 | 0.1470 | 0.0003 | 0.0036 |
| 01001 | 0.0367 | 0.0638 | 0.0338 | 0.0029 | 0.0300 |
| 01010 | 0.0629 | 0.0340 | 0.0646 | 0.0017 | 0.0306 |
| 01011 | 0.0157 | 0.0173 | 0.0160 | 0.0003 | 0.0013 |
| 01100 | 0.0003 | 0.0003 | 0.0005 | 0.0002 | 0.0002 |
| 01101 | 0.0026 | 0.0005 | 0.0026 | 0 | 0.0021 |
| 01110 | 0.0005 | 0.0030 | 0.0004 | 0.0001 | 0.0026 |
| 01111 | 0.0047 | 0.0052 | 0.0048 | 0.0001 | 0.0004 |
| 10000 | 0.0384 | 0.0383 | 0.0371 | 0.0013 | 0.0012 |
| 10001 | 0.0096 | 0.0180 | 0.0093 | 0.0003 | 0.0087 |
| 10010 | 0.0165 | 0.0083 | 0.0167 | 0.0002 | 0.0084 |
| 10011 | 0.0041 | 0.0061 | 0.0045 | 0.0004 | 0.0016 |
| 10100 | 0 | 0.0001 | 0 | 0 | 0.0001 |
| 10101 | 0.0004 | 0 | 0.0004 | 0 | 0.0004 |
| 10110 | 0.0001 | 0.0004 | 0 | 0.0001 | 0.0004 |
| 10111 | 0.0008 | 0.0006 | 0.0006 | 0.0002 | 0 |
| 11000 | 0.0160 | 0.0151 | 0.0178 | 0.0018 | 0.0027 |
| 11001 | 0.0040 | 0.0064 | 0.0046 | 0.0006 | 0.0018 |
| 11010 | 0.0068 | 0.0033 | 0.0073 | 0.0005 | 0.0040 |
| 11011 | 0.0017 | 0.0023 | 0.0020 | 0.0003 | 0.0003 |
| 11100 | 0.0001 | 0.0002 | 0 | 0.0001 | 0.0002 |
| 11101 | 0.0005 | 0.0001 | 0.0008 | 0.0003 | 0.0007 |
| 11110 | 0.0001 | 0.0004 | 0.0001 | 0 | 0.0003 |
| 11111 | 0.0009 | 0.0005 | 0.0016 | 0.0007 | 0.0011 |

QJP represents Quantum Joint Probability, and CJP represents Classical Joint Probability which has been obtained using forward sampling.