qBN - README

General Information

Authors:

- Thierry Rioual
- Tibor Dubois
- Mehmet Gunes

Encadré par: Pierre-Henri WUILLEMIN

Type de document: README

Date de publication: 2024/06/21

Overview

The `qBN` package is designed for representing and performing inference on Bayesian Networks using Quantum Circuits. It leverages the principles of quantum information theory to implement quantum versions of classical probabilistic reasoning algorithms, enabling potentially significant speedups for certain types of computations.

Structure

The package consists of two main classes:

1. **qBNRejection**: This class performs inference via rejection sampling from a Quantum Circuit representation of a Bayesian Network.
2. **qBNMC**: This class constructs a Quantum Circuit representation of a Bayesian Network.

The repository is organized as follows:

```
.
├── Documentation
│   ├── ReadTheDocs
│   └── Theory
├── qBN
│   ├── __init__.py
│   ├── __pycache__
│   ├── qBNMC.py
│   └── qBNRejection.py
├── README.md
├── runTest.py
├── tests
│   ├── __init__.py
│   ├── __pycache__
│   ├── qBNMCTest.py
│   └── qBNRejectionTest.py
├── tree.txt
├── tutorials
│   ├── alarm.dsl
│   ├── alarm.ipynb
│   ├── asia.bif
│   ├── asia.ipynb
│   ├── bayes_nets
│   ├── borujeni.ipynb
│   ├── comparison.ipynb
│   └── tutorial.ipynb
└── XPs
    ├── alarm.dsl
    ├── asia.bif
    ├── asia_plot.py
    ├── comparaison.ipynb
    ├── __init__.py
    ├── plots
    ├── __pycache__
    └── qBNRT.py
```

Implementation

qBNMC

The `qBNMC` class is responsible for building the quantum circuit that represents the Bayesian Network. This involves mapping the variables and their states in the Bayesian Network to qubits and their corresponding quantum states. The quantum circuit is constructed in a way that reflects the probabilistic relationships encoded in the Bayesian Network.

qBNRejection

The `qBNRejection` class utilizes the quantum circuit constructed by `qBNMC` to perform inference. This is done using a process called rejection sampling, where samples are generated from the quantum circuit and then used to compute the probabilities of different states in the Bayesian Network given some evidence.

Research Basis

The implementation of this package is based on the following research papers:

1. **Quantum Inference on Bayesian Networks** by Guang Hao Low: This paper discusses the theory and methods for performing quantum inference on Bayesian Networks.

2. **Quantum circuit representation of Bayesian networks** by Sima E. Borujeni: This paper outlines the methodology for constructing quantum circuits that represent Bayesian Networks.

Usage

To use the `qBN` package, follow these steps:

1. **Install the required dependencies**:

```
pip install qiskit pyAgrum
```

2. **Steps to Follow**

```
# Step 1 : Import the Necessary Libraries
from qBN.qBNMC import qBNMC
from qBN.qBNRejection import qBNRejection
from XPs.qBNRT import qRuntime
import pyAgrum as gum
import pyAgrum.lib.notebook as gnb
import matplotlib.pyplot as plt
from qiskit_ibm_runtime import QiskitRuntimeService

# Step 2 : Initialise the Qiskit Service
service = QiskitRuntimeService(
   'ibm_quantum',
   'ibm-q/open/main',
   'YOUR_IBM_QUANTUM_API_TOKEN'
)
backend = service.get_backend("ibm_brisbane")


# Step 3 : Charge and visualise the Bayesian Network
bn = gum.loadBN("PATH_TO_BN.bif")
#bn : pyAgrum.BayesNet
gnb.showBN(bn, size=20)

# Step 4 : Build the Quantum Circuit
qbn = qBNMC(bn)
#bn : pyAgrum.BayesNet
qc = qbn.buildCircuit(add_measure=True)

# Step 5 : Execution of the Rejection Sampling Method on the Quantum Circuit
qinf = qBNRejection(qbn)
qrt = qRuntime(qinf, backend)
# Define the proof
evidence: Dict[str, int] = {'A': 1, 'B': 0}
```

```
# Define the max number of iterations
max_iter: int = 1000
qinf.setEvidence(evidence)
qinf.setMaxIter(max_iter)
qinf.makeInference(verbose=1)

# Step 6 : Calculate the time of execution and store these values
qinf_run_time = qrt.rejectionSamplingRuntime()
qinf_max_error=(qinf.posterior(target).toarray()-ie.posterior(target).toarray()).max()

print(f"QS - Run time: {qinf_run_time}, Max Error: {qinf_max_error}")
qinf_rt_list.append(qinf_run_time)
qinf_me_list.append(qinf_max_error)

# Step 7 : Visualise the data
# Scatter plot of the quantum inference data we obtained
plt.scatter(ev_prob_list, qinf_rt_list, color="tab:blue", label="QI Execution Time")
# Use a logarithmic measure for axis Y
plt.yscale('log')
plt.xlabel('Probability of the Evidence')
plt.ylabel('Execution Time')
plt.legend()
plt.show()
```

Testing

In the main directory, the folder `tutorials` has a Jupyter-Notebook : `asia.ipynb`, that illustrates an exhaustive example of using all the functions from the package on the ASIA Bayesian Network. This Bayesian Network is a standard test network provided by the pyAgrum library.

Running the Tests

To run the tests, you can execute the `test_qBN.py` script. This will demonstrate the full functionality of the `qBN` package, including each and every methods that is contained in qBNMC and qBNRejection.

Conclusion

The `qBN` package provides a powerful tool for leveraging quantum computing in probabilistic reasoning tasks. By integrating Bayesian Networks with quantum circuits, it opens up new possibilities for faster and more efficient inference, especially for large and complex networks. For detailed information on the implementation and theoretical background, please refer to the research papers references below.

Authors

- [Dubois Tibor, Rioual Thierry, Gunes Mehmet]

References

- Sima E. Borujeni, “Quantum circuit representation of Bayesian networks”
- Guang Hao Low, Theodore J. Yoder, Isaac L. Chuang, “Quantum Inference on Bayesian Networks”
- Documentation de pyAgrum
- Documentation de Qiskit