Informatique Quantique - Aléatoire

Michaël Rollin

25 août 2020

1 Aléatoire

La nature quantique des qubits permet la création de nombres aléatoires. En effet, dans un état parfaitement superposé lors d'une mesure un qubit à 50% de chance d'être à 0 et 50% de chance d'être à 1. A contrario même dans un environnement ayant de la décohérence, celle-ci peut créer du bruits et orienter les chances d'avoir 0 ou 1 en sortie mais celui-ci n'est actuellement pas contrôlable et sa variabilité dépend du niveau d'isolation des qubits.

Dé quantique Voici l'exemple d'un lancement de dé, j'utilise un simulateur me permettant de lancer mon dé hors ligne. Le simulateur émet lui également un minimum de bruit mais celui-ci est négligeable.

```
from qiskit import Aer, QuantumCircuit, execute, IBMQ
from qiskit.tools.monitor import job_monitor

# Creation d'un circuit de 5 qubits.
nb_qubits = 5
circ = QuantumCircuit(nb_qubits)

# Application d'une porte de Hadamart afin de les mettre en etat de superposition
for i in range(0, nb_qubits):
    circ.h(i)

# Ajout des sondes de mesures
meas = QuantumCircuit(nb_qubits, nb_qubits)
# Mapping qubit // bit
meas.measure(range(nb_qubits), range(nb_qubits))

qc = circ+meas
print(qc)
```

```
# Initialisation du simulateur parfait (Qasm)
qasm = Aer.get_backend('qasm_simulator')

# Initialisation de la simulation sur un vrai ordinateur
IBMQ.load_account()
provider = IBMQ.get_provider('ibm-q')
quantum_computer = provider.get_backend('ibmq_burlington')

# Lancement de la simulation
backend_sim = qasm  # Choisi ton simulateur : <quantum_computer> or <qasm>

# On precise que l'on souhaite n'avoir qu'un shot
# et qu'on veut garder le resultat en memoire.
job = execute(qc, backend_sim, shots=1, memory=True)
# Utiliser le job ci-dessous en cas de lanceent sur un véritable ordinateur quantique.
# job = execute(qc, backend_sim, shots=1, optimization_level=3, memory=True)
job_monitor(job)
```

```
# Resultats
result_sim = job.result()
```

```
memory = result_sim.get_memory()

result = int(memory[0], 2)  # Conversion du binaire en integer

# Conversion du nombre en tirage  de 1 a 20
value_dice = round((result + 1) / 2**nb_qubits * 20)
if value_dice == 0:
    value_dice += 1

print("De : ", value_dice)
```

Le code complet et les futur évolution sont disponibles ici : GitHub.