Université de Corse

Année universitaire 2024-2025

Rapport de Séminaire

L'algorithme quantique

Présenté par :
Léandre RAETH, Thomas SANNA

Séminaire présenté le 03/04/2025

L3 Sciences et Technologies, Parcours Informatique

Résumé

Ce rapport de séminaire vise à vulgariser le sujet de l'algorithme quantique pour un public non-initié. Nous expliquons les concepts fondamentaux de l'informatique quantique, en mettant l'accent sur la génération de nombres aléatoires, et présentons un exemple de code utilisant le framework Qiskit. Nous discutons également des défis actuels et des perspectives futures de l'informatique quantique.

Table des matières

1 Introduction

1.1 Présentation du sujet

L'informatique quantique est une discipline émergente qui promet de révolutionner la manière dont nous traitons l'information. Contrairement à l'informatique classique, qui utilise des bits pour représenter les données sous forme de 0 et de 1, l'informatique quantique utilise des qubits, qui peuvent exister simultanément dans plusieurs états grâce aux principes de superposition et d'intrication quantiques que nous expliquerons plus tard dans ce rapport. Cette capacité unique permet aux ordinateurs quantiques de résoudre certains problèmes, assez spécifiques, beaucoup plus rapidement que les ordinateurs classiques.

L'objectif de ce rapport de séminaire est d'avant tout de vulgariser le sujet sur l'algorithme quantique pour un public non-initié. Nous allons expliquer les concepts fondamentaux de l'informatique quantique, en mettant l'accent sur la génération de nombres aléatoires, qui est un exemple simple, mais très important, d'application de l'informatique quantique. Nous allons également présenter un exemple de code utilisant le framework Qiskit pour générer des nombres aléatoires en informatique quantique.

Ce rapport est structuré comme suit : nous commencerons par une comparaison entre l'informatique classique et quantique, suivie d'une explication des propriétés fondamentales de l'informatique quantique. Ensuite, nous aborderons la génération de nombres aléatoires en informatique quantique et présenterons un exemple de code utilisant le framework Qiskit. Enfin, nous discuterons des défis actuels et des perspectives futures de l'informatique quantique avant de conclure par une session de questions-réponses.

Nous avons choisi ce sujet car l'informatique quantique représente une frontière fascinante de la science et de la technologie, avec le potentiel de transformer de nombreux domaines. On espère que ce rapport vous donnera un aperçu clair et concis de cette technologie encore émergente.

1.2 Relation entre informatique quantique et physique quantique

Une petite parenthèse sur la relation entre l'informatique quantique et la physique quantique semble importante pour recontextualiser le sujet.

L'informatique quantique est une discipline interdisciplinaire qui se situe à l'intersection de l'informatique et de la physique quantique. Bien qu'elle ne soit pas un sous-domaine de la physique quantique, elle repose sur les principes fondamentaux de cette dernière pour fonctionner.

En effet, la physique quantique est une branche de la physique qui étudie les phénomènes à l'échelle atomique et subatomique. On y étudie un tout autre monde avec des propriétés uniques par rapport à la physique classique, comme la superposition, l'intrication et l'interférence quantiques. Ces propriétés sont à la base de l'informatique quantique, qui utilise des qubits pour stocker et manipuler l'information.

Ainsi, l'informatique quantique est une application pratique de la physique quantique, qui permet de résoudre des problèmes complexes plus rapidement que les ordinateurs classiques. C'est pourquoi il est important de comprendre les concepts de la physique quantique pour comprendre l'informatique quantique.

2 Informatique classique et Informatique quantique

2.1 Bits classiques

En informatique classique, l'unité de base est le **bit** :

$$0 \text{ ou } 1$$

En réalité, il correspond à la mesure d'**absence** (0) ou de **présence** (1) d'électricité dans les transistors de nos ordinateurs classiques.

Mis à la suite, ces bits forment alors un langage complet, un système binaire qui permet de transmettre une vaste quantité d'informations différentes.

Avec une série de bits, on peut encoder tous types de nombres :

$$168_{10} \rightarrow 10101000_2$$

Et les booléens essentiels aux portes logiques :

$$1 \rightarrow \text{True} \quad | \quad 0 \rightarrow \text{False}$$

Avec ces blocs fondamentaux, nous pouvons alors presque décrire tout ce que l'on souhaite :

- Une couleur : 3 valeurs de vert, bleu et rouge situées entre 0 et 255.
- Les caractères avec lesquels sont écrits ce rapport : un nombre représentant chacun des symboles.
- Une position x, y de votre souris.

Et surtout : effectuer des **calculs**, d'abord simples, puis une suite de calculs plus complexes, sous forme d'algorithmes. C'est dans ce domaine que le bit quantique va pouvoir être utile et surpasser le bit classique.

2.2 Qubits quantiques

En informatique quantique, l'unité de base est le **qubit**. Contrairement au bit classique qui peut être soit 0 soit 1, un qubit peut exister dans une superposition des deux états, **0 et 1 simultanément**. QUBITS 2025 Ce concept peut sembler étrange au premier abord... et il ne l'est pas moins par la suite ! Mais il constitue la base de l'informatique quantique et nous allons essayer de comprendre comment il permet d'ouvrir la voie à des calculs bien plus puissants que ceux possibles avec des ordinateurs classiques.

Un bit quantique n'est alors plus la mesure d'un courant mais la mesure d'un **phénomène physique** tel que le spin d'une particule ou la polarisation d'un photon.

L'informatique quantique repose sur quatre principes clés : **Superposition**, **Intrication**, **Décohérence**, **Interférence**. IBM 2024 Nous allons explorer chacun de ces principes en détails, pour le moment, commençons par les présentations :

1. **Superposition** : Un qubit peut être dans un état de 0 et 1 en même temps, ce qui lui permet de réaliser plusieurs calculs simultanément.
2. **Intrication** : Deux qubits peuvent être entrelacés de sorte que l'état de l'un dépend instantanément de l'état de l'autre, même à distance.
3. **Décohérence** : Les qubits sont extrêmement sensibles à leur environnement, ce qui peut provoquer une perte d'information (c'est ce qu'on appelle la décohérence). Cela représente un des défis majeurs pour les ordinateurs quantiques.
4. **Interférence** : Les états des qubits peuvent interférer entre eux, ce qui permet d'amplifier certaines probabilités et d'annuler d'autres, optimisant ainsi le calcul.

2.3 Comparaison

Bits classiques	**Qubits quantiques**
0 ou 1	0, 1, ou une superposition des deux états (0 et 1 en même temps)
Présence ou absence d'électricité	Superposition d'états grâce à des phénomènes quantiques
Utilise des portes logiques classiques (ex : AND, OR)	Utilise des portes quantiques (ex : Hadamard, CNOT) qui exploitent la superposition et l'entrelacement
Ne peut effectuer qu'un calcul à la fois	Peut effectuer plusieurs calculs simultanément grâce à la superposition
La mesure donne un résultat définitif : 0 ou 1	La mesure "effondre" le qubit dans l'un des deux états avec une probabilité
Calculs classiques, affichage, stockage de données	Résolution de problèmes complexes (cryptographie, optimisation, simulation moléculaire)

Table 1 – Comparaison entre bits classiques et qubits quantiques

3 Propriétés fondamentales de l'informatique quantique

3.1 Superposition

Commençons donc par la superposition, cette fameuse idée de qubit dans des états de 0 et 1 simultanés.

Un qubit est défini par :

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Où α et β sont des amplitudes complexes qui déterminent les probabilités d'obtenir respectivement 0 ou 1 lors d'une mesure, α et β vérifiant $|\alpha|^2 + |\beta|^2 = 1$ (probabilités des états 0 et 1).EvidenCorte 2024 C'est la représentation mathématique d'un qubit en notation bra-ket (notation de Dirac). (On utilise la notation $|0\rangle$ et $|1\rangle$ au lieu de juste 0 et 1 pour bien différencier qu'on parle d'un état quantique et pour utiliser certaines propriétés mathématiques et physiques).

Par exemple, l'état $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ signifie que le qubit a 50% de chances d'être mesuré dans l'état $|0\rangle$ et 50% de chances d'être mesuré dans l'état $|1\rangle$: $\left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$.

Les probabilités de mesurer $|0\rangle$ ou $|1\rangle$ sont bien égales à $\frac{1}{2}$.

Cet état de superposition est créé à l'aide d'une porte de Hadamard, qui permet de générer une superposition, nous y reviendrons un peu plus tard.

Grâce à la superposition, un processeur quantique peut explorer plusieurs solutions en parallèle, contrairement à un ordinateur classique qui traite les données séquentiellement. Un processeur quantique ne donne cependant pas toutes les solutions instantanément : il faut encore une mesure finale pour obtenir un résultat exploitable.

Concrètement, cela permet d'effectuer certaines opérations sur tous ces états en parallèle, au lieu de les traiter un par un : Imaginons que l'on cherche le bon code à 4 chiffres pour ouvrir un cadenas.

- Un ordinateur classique va essayer 0000, 0001, 0002, ..., 9999 un par un jusqu'à trouver le bon. (Même si quelques optimisations sont possibles)
- Un ordinateur quantique peut tester toutes les combinaisons en même temps, puis utiliser un algorithme spécial pour trouver la bonne bien plus vite.

3.2 Mesure

Lorsqu'on mesure un qubit, il **se fige dans un état classique** (0 ou 1) avec une certaine probabilité.

$$P(0) = |\alpha|^2$$
$$P(1) = |\beta|^2$$

Après la mesure, la superposition disparaît.

Cette mesure dépend de la méthode utilisée : le qubit lui-même, physiquement, peut être représenté selon différentes méthodes (atomes, ions, supraconducteurs).Microsoft 2025bStackExchange 2019

3.3 Intrication (Entanglement)

Deux qubits peuvent être intriqués, ce qui signifie que leur état est corrélé, même s'ils sont séparés par des milliers de kilomètres.

L'explication physique relève de la physique quantique, mais le phénomène est bien réel et est utilisé pour permettre le parallélisme quantique : la capacité des ordinateurs quantiques à effectuer plusieurs calculs simultanément. Ainsi, les ordinateurs quantiques manipulent plusieurs qubits en une seule opération, au lieu de manipuler chaque qubit individuellement.

Par exemple, considérons deux qubits qui sont initialement préparés dans un état intriqué. Si une mesure est effectuée sur l'un des qubits et que l'on découvre qu'il est dans l'état $|0\rangle$, alors l'état de l'autre qubit s'effondre immédiatement dans l'état $|0\rangle$ aussi. De même, si le premier qubit est mesuré dans l'état $|1\rangle$, l'état du second qubit s'effondre également dans l'état $|1\rangle$.Microsoft 2025a

3.4 Porte quantique et calcul quantique

Les qubits sont manipulés à l'aide de **portes quantiques**.

Pour rappel, en informatique classique, nous avons les portes :

Porte NOT	**Porte AND**	**Porte OR**
Inverse l'état du bit	Renvoie 1 seulement si les deux entrées sont 1	Renvoie 1 si au moins une des entrées est 1

Table 2 – Portes logiques classiques

Et autres combinaisons. Ce sont ces portes qui permettent de faire des calculs logiques, des opérations d'addition et bien d'autres dans nos ordinateurs.

Il existe leurs équivalents en informatique quantique :

Porte Hadamard (H)	**Porte CNOT**
Crée une superposition	Crée l'intrication entre qubits

Table 3 – Comparaison des portes Hadamard et CNOT

Détailons leur fonctionnement :

3.4.1 Porte Hadamard (H)

La porte Hadamard est l'une des portes fondamentales en informatique quantique.

La porte Hadamard transforme un qubit dans l'état $|0\rangle$ en une superposition égale des états $|0\rangle$ et $|1\rangle$, avec une probabilité de 50% pour chaque. Par exemple, si un qubit est dans l'état $|0\rangle$, après l'application de la porte Hadamard, il sera dans l'état :

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

Physiquement, cette porte est réalisée avec des impulsions électromagnétiques dans un ordinateur quantique supraconducteur.

La porte Hadamard perturbe le qubit en le mettant dans un état instable tant qu'on ne le mesure pas.

C'est comme jeter une pièce en l'air : tant qu'elle tourne, elle n'est ni pile ni face, mais une combinaison des deux.

3.4.2 Porte CNOT

La porte CNOT agit sur deux qubits :

— Si le qubit de contrôle est $|0\rangle$, rien ne change.
— Si le qubit de contrôle est $|1\rangle$, le qubit cible est inversé ($|0\rangle$ devient $|1\rangle$ et inversement).

Exemple d'utilisation : On commence avec $|00\rangle$. Après une porte Hadamard sur le premier qubit, l'état devient :

$$\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$$

Puis, la porte CNOT inverse le deuxième qubit si le premier est $|1\rangle$, donnant :

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Lors de la mesure du premier qubit :

— Si $|0\rangle$, le second est aussi $|0\rangle$.
— Si $|1\rangle$, le second est $|1\rangle$.

Les qubits sont intriqués, avec une corrélation instantanée. En bref, comme en informatique classique, on utilise ces "portes" pour manipuler les données (bits ou qubits) rentrantes et appliquer des calculs dessus. En informatique quantique, on manipule pour ensuite faire des calculs sur les probabilités.

	Porte de Hadamard	**Porte CNOT**
Entrée :	$\|0\rangle$	$\|AB\rangle$
Sortie :	$\frac{1}{\sqrt{2}}(\|0\rangle + \|1\rangle)$	Si $A = 0 \rightarrow \|AB\rangle$ Si $A = 1 \rightarrow \|A\overline{B}\rangle$

3.5 Interférence

L'interférence quantique se produit lorsque les amplitudes de probabilité des différents états d'un système quantique se combinent. Selon la manière dont elles se combinent, les probabilités de certains résultats peuvent être amplifiées (interférence constructive) ou diminuées (interférence destructive). C'est un effet clé pour exploiter pleinement la puissance des ordinateurs quantiques, car il permet d'augmenter les chances de certains résultats tout en réduisant celles des autres.

4 Exemple d'application : Génération de Nombre aléatoire

4.1 Problématique

Lors de la génération d'un nombre aléatoire en informatique traditionnelle, par exemple, en Python avec le module 'random()', le nombre généré est en réalité dit pseudo-aléatoire car l'algorithme de Mersenne Twister repose sur une graine pour initialiser le générateur de nombres aléatoires. Cette graine permet de prédire le prochain nombre pseudo-aléatoire, ce qui constitue un problème important pour de nombreux projets confidentiels, y compris la cryptographie.

Pour les projets impliquant une confidentialité sévère, comme la cryptographie, l'un des plus grands problèmes avec la génération de nombres pseudo-aléatoires est la capacité à produire des nombres élevés qui sont extrêmement difficiles à prédire. Par exemple, lors de la génération du sel du mot de passe afin de le hacher, un générateur de nombres aléatoires est utilisé. Si ce générateur est prévisible, un hacker compétent trouverait facilement le mot de passe en un rien de temps.

C'est pourquoi il est important de s'assurer que la graine choisie permettra un certain degré d'imprévisibilité.

En revanche, en informatique quantique, la génération de nombres aléatoires peut être véritablement aléatoire grâce aux propriétés quantiques expliquées il y a peu de temps. En effet, en utilisant un ordinateur quantique, on peut mesurer l'état d'un qubit en superposition pour obtenir un résultat aléatoire non-déterministe !

4.2 Fonctionnement

Pour comprendre comment fonctionne la génération de nombres aléatoires en informatique quantique, il faut se pencher sur les propriétés des qubits. Un qubit, comme on l'a vu plus tôt, peut être dans un état de superposition, ce qui signifie qu'il peut représenter simultanément les états 0 et 1. Lorsqu'on mesure un qubit en superposition, le résultat de la mesure est complètement aléatoire entre 0 et 1.

— **Préparation du qubit** : On commence par préparer un qubit dans un état de superposition. Cela peut être réalisé en appliquant une porte Hadamard (H) à un qubit initialement dans l'état $|0\rangle$. La porte Hadamard transforme l'état $|0\rangle$ en une superposition égale des états $|0\rangle$ et $|1\rangle$:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

La porte d'Hadamard est extrêmement utilisée en informatique quantique Ces superpositions ne sont pas des simulations de probabilités, mais des états réels qui peuvent être mesurés.

— **Mesure du qubit** : Une fois le qubit en superposition, on procède à sa mesure. La mesure d'un qubit en superposition donne un résultat

aléatoire, soit 0 soit 1, avec une probabilité de 50% pour chaque état. Ce processus est intrinsèquement aléatoire et ne peut pas être prédit, même si l'on connaît l'état initial du qubit.

— **Génération de séquences aléatoires** : En répétant ce processus de préparation et de mesure de qubits en superposition `n` fois, on peut générer des séquences de bits aléatoires.

Ainsi, la génération de nombres aléatoires en informatique quantique repose sur les principes fondamentaux de la mécanique quantique, offrant une source de véritable aléa, contrairement aux méthodes pseudo-aléatoires utilisées en informatique classique. Nous pourrons trouver le code de l'algorithme de génération de nombres aléatoires en informatique quantique dans la section suivante.

4.3 Comparaison des méthodes

Algorithme classique	**Algorithme quantique**
Pseudo-aléatoire	Non déterministe
Prédictible	Imprévisible
Basé sur des algorithmes	Basé sur des propriétés quantiques
Temps de calcul rapide	Temps de calcul plus lent

TABLE 4 – Comparaison de la génération de nombres aléatoires en utilisant un algorithme classique et quantique

5 Expérimentation : coder un algorithme quantique

5.1 Introduction

Il existe deux manières de coder un algorithme quantique : en utilisant un simulateur quantique ou un véritable ordinateur quantique. Pour coder sur un vrai ordinateur quantique, il est possible d'utiliser des services cloud comme IBM Quantum Experience, qui permettent d'accéder à des ordinateurs quantiques en ligne. IBMQP 2025

Cependant, pour des raisons de simplicité, nous allons utiliser un simulateur quantique, qui permet de simuler un ordinateur quantique sur un ordinateur classique. Pour cela, nous allons utiliser le langage de programmation Qiskit.

En effet, Qiskit est un framework open-source développé par IBM en 2017 pour la programmation d'algorithme quantique en Python Qiskit 2025. Qiskit permettra d'utiliser cette fameuse porte Hadamard pour préparer un qubit dans un état de superposition.

Il est très important de noter que les simulateurs utilisent des algorithmes pseudo-aléatoires pour mesurer les résultats. En effet, on utilise un algorithme de randomisation entre 0 et 1 pour simuler le résultat de la mesure d'un qubit en superposition.

5.1.1 Les schémas de circuits quantiques

Un circuit quantique est une représentation graphique d'un algorithme quantique. Il est composé de qubits, de portes quantiques et de mesures. Les portes quantiques sont des opérations unitaires qui agissent sur un ou plusieurs qubits. Les mesures permettent de lire l'état d'un qubit.

Porte Hadamard : La porte de Hadamard, on l'a vu plus tôt, est une porte quantique qui permet de mettre un qubit dans un état de superposition. Elle est représentée par la lettre H dans un schéma de circuit quantique. Elle prend un qubit initialement dans l'état $|0\rangle$ (ou $|1\rangle$, peu importe, cela n'affectera pas la porte) et le transforme, à la sortie, en une superposition équilibrée des états $|0\rangle$ et $|1\rangle$.

Figure 1 – Schéma de circuit quantique pour la porte Hadamard (QGate 2025)

Mesure : La mesure d'un qubit permet de lire son état. Elle est représentée par un symbole de balance dans un schéma de circuit quantique. La mesure d'un qubit en superposition donne un résultat aléatoire entre 0 et 1, avec une

probabilité de 50% pour chaque état. Elle prend dont un qubit en entrée et renvoie un bit classique (0 ou 1) en sortie.

Information : *Les deux lignes horizontales représentent un bit classique, tandis qu'une seule ligne représente un qubit.*

FIGURE 2 – Schéma de circuit quantique pour la mesure d'un qubit (QGATE 2025)

Circuit d'un algorithme de nombre aléatoire : Ce circuit quantique décris l'algorithme de génération de nombres aléatoires en informatique quantique.

On travaille ici avec qu'un seul qubit `q` initialisé à l'état $|0\rangle$ ou $|1\rangle$ (conventionnellement, on choisit $|0\rangle$). On applique ensuite une porte Hadamard à ce qubit pour le mettre dans un état de superposition. Enfin, on mesure le qubit pour obtenir un résultat aléatoire entre 0 et 1.

Cette mesure est sortie dans la partie `c` du circuit (`c` signifie *classique*). Cette partie renvoie un bit classique, qui est le résultat de la mesure du qubit. En effet, le `1` à droite de `c` signifie le nombre de bits classiques en sortie. Le `0` à côté de la sortie de la mesure, signifie que le bit mesuré est à l'index numéro 0.

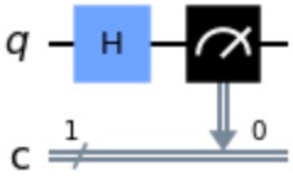

FIGURE 3 – Portes logiques pour la génération d'un nombre aléatoire entre 0 et 1 (SAP 2023)

Circuit avec 8 qubits : Le problème avec le circuit précédent est qu'il ne génère qu'un seul bit aléatoire, ce qui veut dire que le nombre aléatoire généré est soit 0 soit 1. Pour générer un nombre aléatoire plus grand, il suffit de répéter le processus de préparation et de mesure de qubits en superposition plusieurs fois, ou alors d'utiliser plusieurs qubits en superposition en même temps.

Ici, on obtient un circuit quantique pour la génération d'un nombre aléatoire avec 8 qubits. En une seule exécution, ce circuit génère un nombre aléatoire entre 0 et 255 (car $2^8 = 256$).

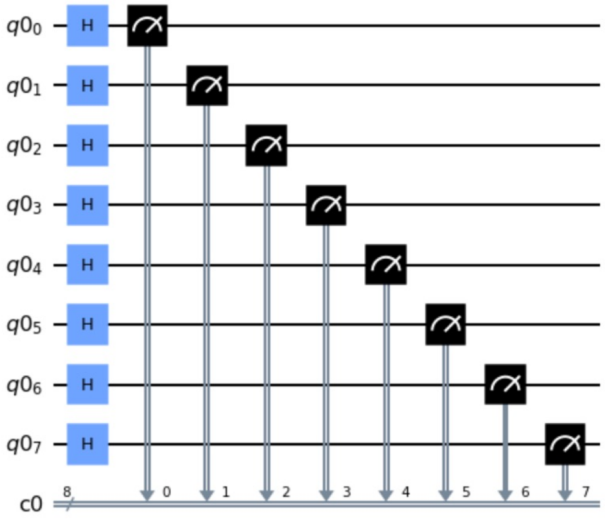

FIGURE 4 – Circuit quantique pour la génération d'un nombre aléatoire avec 8 qubits (SAP 2023)

5.2 Code de l'algorithme

Dans cette partie, il est question de coder l'algorithme de génération de nombres aléatoires avec 8 qubits en informatique quantique en utilisant Qiskit.

5.2.1 Prérequis

Il est étonnament assez simple de coder un algorithme quantique en Python avec Qiskit. En voici les prérequis :

— Installer Python : `https://www.python.org/downloads/`
— Installer Qiskit : `pip install qiskit`
— Installer Qiskit Aer : `pip install qiskit-aer` (AER 2025)

Il est aussi préférable de coder sur Jupyter Notebook, qui permet d'exécuter du code Python en temps réel.

5.2.2 Initialisation du circuit

Le circuit quantique est initialisé avec 8 qubits en entrée et 8 bits classiques en sortie. On applique ensuite une porte Hadamard à chaque qubit pour les mettre dans un état de superposition, comme expliqué précédemment. On mesure ensuite chaque qubit pour obtenir un résultat aléatoire entre 0 et 1, stocké dans leurs bits classiques respectifs. (Par exemple, le résultat du qubit 0 est stocké dans le bit classique 0, et ainsi de suite.)

```
1  from qiskit import QuantumRegister, ClassicalRegister,
     QuantumCircuit, transpile
```

```
from qiskit_aer import Aer

q = QuantumRegister(8, 'q') # initialisation d'un registre quantique de 8 qubits
c = ClassicalRegister(8, 'c') # initialisation d'un registre classique de 8 bits
circuit = QuantumCircuit(q, c) # initialisation d'un circuit quantique avec les registres q et c

circuit.h(q[0]) # porte de Hadamard sur le premier qubit
circuit.h(q[1]) # porte de Hadamard sur le deuxieme qubit
circuit.h(q[2]) # ...
circuit.h(q[3])
circuit.h(q[4])
circuit.h(q[5])
circuit.h(q[6])
circuit.h(q[7])

""" # simplification
for elt_qubit in q:
    circuit.h(elt)
"""

circuit.measure(q, c) # mesure de tous les qubits dans les bits classiques correspondants
```

Listing 1 – Initialisation du circuit avec 8 qubits

5.2.3 Exécution du circuit

Pour exécuter le circuit, on utilise un simulateur quantique. Ici, on utilise le simulateur Aer de Qiskit, qui permet de simuler un ordinateur quantique sur un ordinateur classique.

À la fin de l'execution, on obtient en retour un dictionnaire contenant les résultats de la simulation, c'est-à-dire les nombres aléatoires générés.

```
simulateur = Aer.get_backend('aer_simulator') # initialisation du simulateur Aer avec le backend aer_simulator
circuitCompile = transpile(circuit, simulateur) # compilation du circuit pour le simulateur
result = simulateur.run(circuitCompile, shots=1).result() # shot=1 veut dire qu'on execute le circuit une seule fois

counts = result.get_counts(circuit)
print("\nResultats de la mesure :", counts)
```

Listing 2 – Exécution du circuit

Note additionnelle *: On peut voir dans certains codes sur internet que le simulateur s'appelle* `qasm_simulator` *au lieu de* `aer_simulator`*. En effet,* `qasm_simulator` *est une version nientôt obsolète du simulateur Aer. On conseille aujourd'hui d'utiliser* `aer_simulator` *depuis peu. (*EGRETTATHULA *2022)*

En retour de cette exécution, on obtient un dictionnaire qui comporte le nombre binaire généré :

```
Resultats de la mesure : {'00001010': 1}
```

Listing 3 – Résultat de la mesure : {'00001010': 1}

Ce dictionnaire contient le nombre binaire 8 bits généré, avec sa fréquence d'apparition. Il n'y a qu'une seule clé dans ce dictionnaire, qui est le nombre binaire généré, et la valeur associée est le nombre de fois que ce nombre a été généré (une seule fois car le circuit a été exécuté une fois, CF Listing 2 `"shots=1"`).

Ici, le nombre généré est 00001010, qui correspond à 10 en décimal. On peut donc dire que le nombre aléatoire généré est 10.

5.2.4 Visualisation du circuit

Il est possible de visualiser le circuit quantique généré avec Qiskit. Pour cela, on utilise la méthode `circuit_drawer()` du circuit.

```
from qiskit import QuantumCircuit

circuit_drawer(circuit, output='mpl')
```

Listing 4 – Visualisation du circuit

On obtient bien un schéma de circuit quantique avec les portes Hadamard et les mesures, comme celle vue précédemment.

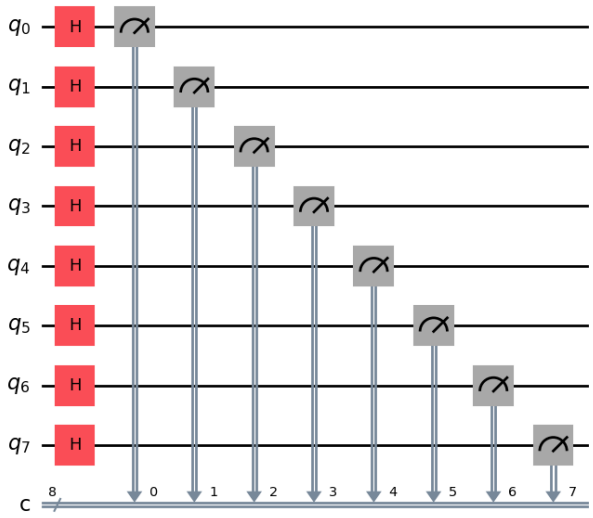

FIGURE 5 – Schéma de circuit quantique pour la génération d'un nombre aléatoire avec 8 qubits, généré à partir de Qiskit

5.3 Conclusion de l'expérimentation

Cette expérimentation est sûrement la plus simple à réaliser et à comprendre pour un débutant en informatique quantique. Elle permet de mettre en pratique les concepts de superposition et de mesure des qubits en quelques lignes de code.

Il n'y a pas grand chose à faire de plus si vous souhaitez essayer sur une vraie machine quantique, il suffit de changer le backend du simulateur Aer pour un vrai ordinateur quantique. Cependant, il est important de noter que les ordinateurs quantiques sont encore en développement et, il peut y avoir en conséquence, des erreurs dans les résultats. Les simulations quantiques offrent une alternative plus fiable pour l'instant, bien qu'un manque de scalabilité soit présent pour des circuits plus complexes, ainsi qu'un manque de réalisme quantique.

6 Limites et perspectives

6.1 Défis actuels

Comme nous avons pu le voir, l'informatique quantique est une technologie prometteuse, offrant même l'espoir de voir l'humanité triompher de problèmes complexes. Mais elle reste confrontée à plusieurs défis majeurs qui ralentissent son développement et son adoption à grande échelle :

1. **Décohérence et correction d'erreurs**
   L'un des problèmes les plus importants est la décohérence quantique, qui entraîne une perte rapide d'information lorsque les qubits interagissent et sont perturbés par leur environnement. Cela limite le temps de calcul utile et nécessite des techniques avancées de correction d'erreurs quantiques. Cependant, les codes correcteurs actuels nécessitent des dizaines, voire des centaines de qubits physiques (qui ne sont pas utilisés directement pour le calcul) pour stabiliser un seul qubit logique (utilisé pour le calcul). Nous verrons que de récentes avancées ont été réalisées sur ce sujet.
2. **Scalabilité et fiabilité des qubits**
   La mise à l'échelle des ordinateurs quantiques est un autre défi. Les processeurs quantiques actuels possèdent encore un nombre limité de qubits exploitables. Par exemple, IBM et Google ont construit des processeurs atteignant respectivement 1125* et 105 qubits physiques qui sont encore loin des milliers de qubits stables nécessaires pour des calculs pratiques. (*il est important de noter que la qualité des qubits (comment ils sont gérés, mesurés) est aussi, voir plus importante que leur quantité pour le moment. Tous les qubits de se valent pas !)
3. **Puissance de calcul réelle vs promesses théoriques**
   Si des algorithmes comme celui de (Shor 2025) pour la factorisation ou de Grover pour la recherche dans une base de données sont souvent mis en avant, les gains quantiques restent difficiles à exploiter sur les machines actuelles. Les résultats restent contestés et encore loin d'une application industrielle généralisée.
4. **Besoin en infrastructure cryogénique**
   La plupart des technologies de qubits nécessitent un refroidissement à des températures proches du zéro absolu (15 mK), ce qui augmente les coûts et rend difficile leur intégration dans des systèmes informatiques conventionnels.

6.2 Futur de l'informatique quantique

Malgré ces défis, l'informatique quantique continue de progresser rapidement, avec plusieurs avancées notables qui pourraient accélérer son adoption.

1. **L'essor des qubits topologiques**
   Microsoft travaille activement sur des qubits topologiques, plus résistants aux perturbations et aux erreurs. En 2025, l'entreprise a annoncé avoir

réussi à présenter le Majorana 1, une puce quantique utilisant des qubits topologiques fabriqués à partir de nouveaux matériaux, les topoconducteurs. Un état de la matière qui ouvre la voie à des qubits plus stables et plus performants. Cette avancée pourrait permettre de construire des ordinateurs quantiques dotés de millions de qubits dans un avenir proche, offrant ainsi une fiabilité accrue pour des applications de calculs quantiques à grande échelle. Bolgar 2025

2. **Hybrides quantique-classique**
   Une approche prometteuse consiste à coupler des algorithmes quantiques à des processeurs classiques. Des entreprises comme NVIDIA et Amazon développent des plateformes hybrides permettant d'utiliser des circuits quantiques comme accélérateurs spécialisés pour l'optimisation et l'apprentissage automatique.
3. **Applications concrètes en développement**
   Bien que la cryptographie post-quantique soit un domaine majeur, d'autres secteurs comme la chimie (modélisation de molécules complexes), la finance (optimisation de portefeuilles), et la logistique (optimisation de trajets) commencent à expérimenter des solutions hybrides incluant du calcul quantique.
4. **Démocratisation via le cloud**
   L'accès aux ordinateurs quantiques via le cloud (IBM Quantum, Azure Quantum, AWS Braket) permet à de plus en plus de chercheurs et d'entreprises d'expérimenter et d'optimiser des algorithmes quantiques, facilitant ainsi l'émergence d'applications pratiques.

Si les défis rencontrés la peuvent sembler inquiétant, ces avancés permettent d'apercevoir un futur brillant pour cette discipline. Il faut aussi se rappeler que nous faisions face à des problèmes similaires aux débuts de l'informatique classique :

Les premiers ordinateurs classiques comme l'ENIAC (1945) étaient lents, énergivores et peu fiables. Leur évolution a nécessité des décennies pour atteindre des machines plus rapides et fiables, comme les premiers processeurs à transistors.

Premiers "processeurs" classiques	**Processeurs modernes (i9)**
100 kHz	5,2 GHz (52 000x supérieur)

Table 5 – Évolution des processeurs classiques

De même, l'informatique quantique aujourd'hui est limitée par des qubits fragiles, soumis à la décohérence et au bruit. Mais nous pouvons observer les mêmes avancées majeures qu'il y a 60 ans :

Processeurs	**Année**	**Qubits**	**Quantique vs Classique**
Sycamore (Google)	2019	70	10^4 ans -¿ 200 secondes
Willow (Google)	2024	105	10^{25} ans -¿ 5 min

Table 6 – Évolution des processeurs quantiques

Neven 2024 Sycamore 2024 QProcessors 2025

7 Conclusion et Q&A

C'est toujours très fascinant à entendre parler de l'informatique quantique, mais il est important de garder à l'esprit que nous ne sommes qu'au début de cette technologie. Il reste encore beaucoup de recherche et de développement à faire avant que l'informatique quantique ne devienne une réalité pratique et accessible à tous.

Beaucoup de scientifiques et d'ingénieurs ont, à l'heure actuelle, besoin de se former à cette nouvelle discipline, et il est donc important de continuer à investir dans la recherche et l'éducation pour que l'informatique quantique puisse réaliser son potentiel.

7.1 Session de questions-réponses

- **Q1 : Quelle est la différence entre un ordinateur quantique et un simulateur quantique ?**
- Réponse : Un ordinateur quantique utilise des qubits physiques pour effectuer des calculs basés sur la mécanique quantique, tandis qu'un simulateur quantique utilise un ordinateur classique pour imiter le comportement d'un ordinateur quantique. Les simulateurs sont utiles pour le développement et le test d'algorithmes quantiques, mais ils ne peuvent pas reproduire pleinement les avantages de la véritable computation quantique.
- **Q2 : Quels sont les principaux langages de programmation utilisés pour développer des algorithmes quantiques ?**
- Réponse : Les principaux langages de programmation pour les algorithmes quantiques incluent Qiskit (Python) développé par IBM, Cirq (Python) développé par Google, et Q# développé par Microsoft.
- **Q3 : Comment les ordinateurs quantiques peuvent-ils améliorer la cryptographie ?**
- Réponse : Les ordinateurs quantiques peuvent briser les systèmes de cryptographie actuels basés sur la factorisation de grands nombres (comme RSA) en utilisant des algorithmes comme celui de Shor (Shor 2025). Ils peuvent également renforcer la cryptographie en permettant la création de clés de chiffrement véritablement aléatoires et en facilitant la mise en œuvre de la cryptographie quantique.
- **Q4 : Quels sont les défis liés à la correction d'erreurs dans les ordinateurs quantiques ?**
- Réponse : La correction d'erreurs quantiques est un défi majeur en raison de la sensibilité des qubits à leur environnement, ce qui peut entraîner des erreurs de décohérence et de bruit. Les codes de correction d'erreurs quantiques nécessitent un grand nombre de qubits physiques pour protéger un seul qubit logique.
- **Q5 : Comment les ordinateurs quantiques peuvent-ils être utilisés dans l'optimisation ?**
- Réponse : Les ordinateurs quantiques peuvent résoudre des problèmes

d'optimisation complexes plus efficacement que les ordinateurs classiques en utilisant des algorithmes quantiques comme l'algorithme de Grover (Grover 2025). Par exemple, pour trouver un élément dans une liste de n éléments, un algorithme classique a une complexité de $O(n)$, tandis qu'un algorithme quantique a une complexité de $O(\sqrt{n})$, ce qui est beaucoup plus rapide.

— **Q6 : Quels sont les principaux obstacles à l'adoption généralisée des ordinateurs quantiques ?**
— Réponse : Les principaux obstacles incluent la décohérence, le bruit des qubits, la mise à l'échelle des systèmes, le besoin en infrastructure cryogénique, et le développement de logiciels et d'algorithmes adaptés. De plus, il est nécessaire de former des experts en informatique quantique.
— **Q7 : Qu'avez-vous appris de cette présentation ?**
— Réponse : Nous nous intéressions déjà pas mal à l'informatique quantique, mais cela nous a permis de mieux comprendre les concepts de base suivant les mathématiques et la physique quantique. Coder un algorithme quantique était une première pour nous.
— **Q8 : Que pensez-vous de l'informatique quantique ?**
— Réponse : Ce sujet nous fait comprendre assez rapidement que nous vivons encore aux touts débuts de l'informatique quantique et qu'il nécessite encore beaucoup de recherche et de développement pour exploiter pleinement son potentiel. Cependant, nous sommes très enthousiastes à l'idée de voir comment cette technologie va évoluer dans les années à venir et comment elle pourrait transformer notre façon de traiter l'information.

8 Annexes

8.1 Glossaire

- **Algorithme de Grover** : Algorithme quantique pour la recherche dans une base de données non structurée. (Grover 2025)
- **Algorithme de Shor** : Algorithme quantique pour la factorisation de grands nombres, qui menace la cryptographie classique. (Shor 2025)
- **Cirq** : Framework open-source développé par Google pour la programmation d'algorithmes quantiques.
- **Correction d'erreurs quantiques** : Techniques utilisées pour protéger les qubits contre les erreurs causées par la décohérence et le bruit.
- **Décohérence** : Perte d'information quantique due à l'interaction avec l'environnement.
- **IBM Quantum Experience** : Plateforme cloud d'IBM permettant d'accéder à des ordinateurs quantiques.
- **Intrication** : Propriété quantique où deux qubits sont liés de telle manière que l'état de l'un dépend de l'état de l'autre.
- **Mesure** : Processus d'obtention d'un résultat classique à partir d'un qubit en superposition.
- **Porte quantique** : Opération qui agit sur un ou plusieurs qubits pour effectuer des calculs.
- **Q#** : Langage de programmation quantique développé par Microsoft.
- **QASM** : Quantum Assembly Language, un langage de bas niveau pour représenter des circuits quantiques.
- **Qiskit** : Framework open-source pour la programmation d'algorithmes quantiques en Python.
- **Qubit** : Unité de base de l'information quantique, analogue au bit classique.
- **Superposition** : État dans lequel un qubit peut représenter simultanément plusieurs valeurs.
- **Topoconducteurs** : Matériaux utilisés pour fabriquer des qubits topologiques, offrant une meilleure stabilité et performance.

8.2 Liens utiles

- **Cirq** : `https://quantumai.google/cirq`
- **Google Quantum AI** : `https://quantumai.google/`
- **IBM Quantum Experience** : `https://quantum-computing.ibm.com/`
- **Microsoft Quantum Development Kit** : `https://azure.microsoft.com/en-us/services/quantum-development-kit/`
- **Q#** : `https://docs.microsoft.com/en-us/quantum/`
- **QASM** : `https://www.quantum-inspire.com/kbase/cqasm/`
- **Qiskit** : `https://qiskit.org/`
- **Stack Exchange** : `https://quantumcomputing.stackexchange.com/`
- **Wikipedia** : `https://fr.wikipedia.org/wiki/Informatique_quantique`

8.3 L'utilisation de l'intelligence artificielle pour la recherche et la rédaction

L'intelligence artificielle peut se sembler extrêmement utile pour obtenir des informations assez précises sur certains domaines de ce sujet.

Nous avons en effet utiliser l'intelligence artificielle pour, par exemple, avoir la confirmation du fait qu'un un simulateur quantique utilisait bien un algorithme pseudo-aléatoire pour simuler la mesure d'un qubit.

Voici quelques une des questions posées à l'IA :

- *Comment un simulateur quantique arrive à creer des qubits/porte d'Hadamarde ?*
- *Dans cet exemple (expliqué plus tôt par l'IA), comment mesure-t-on ce qubit ?*
- *Si l'on fixe une seed, et que l'on a trois qubits initialisés à 0 et un initialisé à 1. Est-ce qu'apres la porte d'Hadamarde, le bit initialisé à 1 sera forcément different de ceux à 0 suite à la mesure ? (toujours dans un cadre de simulation)*
- *À quel point l'enjeu d'avoir un ordinateur quantique est important dans la création d'un nombre purement aléatoire ? Recherche, si jamais, des interview, etc...*
- *Existe-t-il d'autres domaines qui nécessitent d'autant plus les ordinateurs quantique ?*

La recherche par image a aussi été utilisée pour obtenir une description et explication plus détaillée sur les circuits quantiques

Sur le contenu de ce rapport, l'IA a pu juger de la pertinence de certaines phrases, et de la formulation de celles-ci. Nous doutions aussi du fait que nous parlions un peu trop de la physique quantique et pas assez de l'algorithme quantique, l'IA a pu nous suggérer des changements de contenus pour rééquilibrer le rapport.

Nous avons aussi pu utiliser l'IA pour rallonger certains paragraphes, en lui demandant de reformuler certaines phrases, ou de les développer.

L'IA a permis de corriger certaines fautes d'orthographes, de certains problèmes de compilation en Latex, et de nous aider à écrire certaines équations.

L'aide à la mise en page a aussi été très utile, en nous suggérant de mettre certaines parties en gras, ou d'ajouter des titres pour mieux structurer le rapport.

9 Bibliographie

Références

Aer (2025). *Getting started - Qiskit Aer 0.16.1 — qiskit.github.io*. https://qiskit.github.io/qiskit-aer/getting_started.html. [Accédé le 13-03-2025].

Bolgar, Catherine (2025). *Microsoft's Majorana 1 chip carves new path for quantum computing*. https://news.microsoft.com/source/features/innovation/microsofts-majorana-1-chip-carves-new-path-for-quantum-computing/. [Accédé le 21-02-2025].

EgrettaThula (2022). *What are the differences between Qiskit's AerSimulator, QasmSimulator and StatevectorSimulator ? — quantumcomputing.stackexchange.com*. https://quantumcomputing.stackexchange.com/questions/24072/what-are-the-differences-between-qiskits-aersimulator-qasmsimulator-and-statev. [Accédé le 14-03-2025].

EvidenCorte (2024). *Formation Corte/MesoNet - Damien Nicolazic.* . [Accédé le 16-03-2025].

Grover (2025). *Algorithme de Grover — Wikipédia — fr.wikipedia.org*. https://fr.wikipedia.org/wiki/Algorithme_de_Grover. [Accédé le 13-02-2025].

IBM (2024). *Qu'est-ce que l'informatique quantique*. https://www.ibm.com/fr-fr/topics/quantum-computing. [Accédé le 15-03-2025].

IBMQP (2025). *IBM Quantum Platform - Wikipedia — en.wikipedia.org*. https://en.wikipedia.org/wiki/IBM_Quantum_Platform. [Accédé le 13-03-2025].

Microsoft (2025a). *Entanglement*. https://quantum.microsoft.com/en-us/insights/education/concepts/entanglement. [Accédé le 17-03-2025].

— (2025b). *Le qubit en informatique quantique*. https://learn.microsoft.com/fr-fr/azure/quantum/concepts-the-qubit. [Accédé le 16-03-2025].

Neven, Hartmut (2024). *Meet Willow, our state-of-the-art quantum chip*. https://blog.google/technology/research/google-willow-quantum-chip/. [Accédé le 21-03-2025].

QGate (2025). *Porte quantique — Wikipédia — fr.wikipedia.org*. https://fr.wikipedia.org/wiki/Porte_quantique. [Accédé le 13-03-2025].

Qiskit, IBM (2025). *Qiskit - Wikipedia — en.wikipedia.org*. https://en.wikipedia.org/wiki/Qiskit. [Accédé le 13-03-2025].

QProcessors (2025). *List of quantum processors – Wikipedia*. https://en.wikipedia.org/wiki/List_of_quantum_processors. [Accédé le 21-02-2025].

Qubits (2025). *Qubits – Wikipedia*. https://fr.wikipedia.org/wiki/Qubit. [Accédé le 15-03-2025].

SAP (2023). *True randomness with Quantum (Quantum Random number generator) — community.sap.com*. https://community.sap.com/t5/

additional-blogs-by-sap/true-randomness-with-quantum-quantum-random-number-generator/ba-p/13549945. [Accedé le 13-03-2025].

Shor (2025). *Algorithme de Shor — Wikipédia — fr.wikipedia.org*. https://fr.wikipedia.org/wiki/Algorithme_de_Shor. [Accedé le 23-03-2025].

StackExchange (2019). *How are states of a qubit measured in a quantum computer ?* https://quantumcomputing.stackexchange.com/questions/9358/how-are-states-of-a-qubit-measured-in-a-quantum-computer. [Accédé le 16-03-2025].

Sycamore (2024). *Processeur Sycamore – Wikipedia*. https://fr.wikipedia.org/wiki/Processeur_Sycamore. [Accedé le 21-02-2025].