

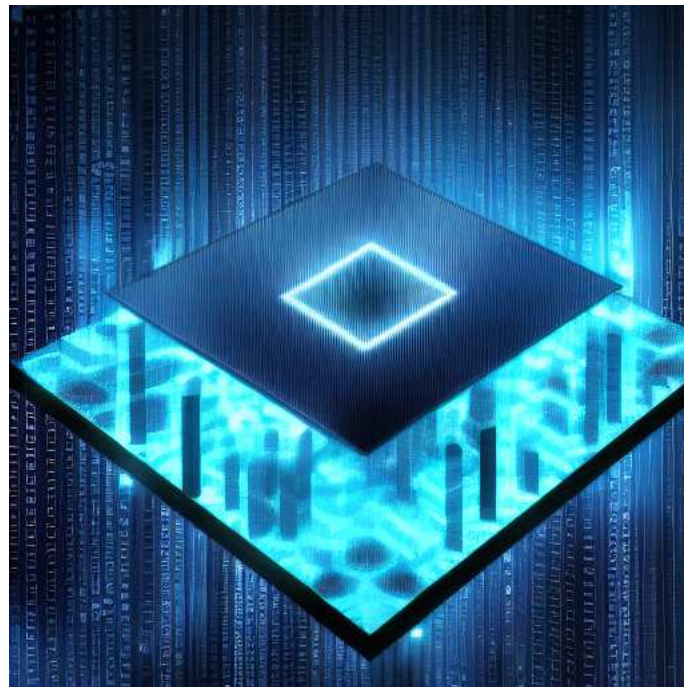


GYMNASE AUGUSTE PICCARD

TRAVAIL DE MATURITÉ

Les algorithmes quantiques

Ou une théorie d'optimisation



Romain BLONDEL
Classe : 3M8

30 octobre 2023

Maître de travail : Frédéric
DE MONTMOLLIN

Abstract

L'ordinateur quantique, un terme qui peut terrifier certains qui voient cela comme de la magie noire, et en rebuter d'autres qui y voient un jouet pour les chercheurs. Néanmoins, il représente pour la plupart des personnes qui s'y intéressent un espoir énorme, une révolution dans le monde de l'informatique qui permettrait de résoudre des problèmes qui sont actuellement trop complexes pour nos ordinateurs classiques.

Même s'ils ne les remplaceront pas, les ordinateurs quantiques vont offrir une accélération dans le calcul de certaines tâches, et permettre de résoudre des problèmes qui sont actuellement trop gros pour nos outils actuels. Ils pourraient aussi réaliser ces calculs avec moins d'énergie que ne le demande un ordinateur classique.

Dans ce travail, nous développerons les bases de l'informatique quantique, par les concepts théoriques nécessaires à la compréhension de ce domaine, mais également par la présentation de quelques algorithmes quantiques qui montrent un avantage par rapport à leurs équivalents classiques, en passant par des résultats qui ont été obtenus sur des ordinateurs quantiques réels et les enjeux dans le futur de cette technologie.

Nous avons essayé de démontrer la beauté de ce domaine, de part son ingéniosité, mais aussi des concepts mathématiques qui sont derrière, et qui sont souvent méconnus du grand public, mais surtout très intéressants et beau dans un sens. Néanmoins, quand ceux-ci sont trop complexes ou rébarbatifs, nous avons essayé de faire comprendre par des aspects intuitifs le fonctionnement de ces algorithmes, en essayant de les mettre en exemples avec des applications concrètes, dans la mesure du possible.

Table des matières

I	Préambule	3
1	Introduction	4
2	Notions théoriques	6
2.1	Informatique	6
2.2	Physique	13
II	Les notions de base	22
3	Un ordinateur classique	23
3.1	Logique	23
3.2	Hardware	27
4	Un qubit	29
4.1	Superposition d'états	29
4.2	Opérations sur un qubit	30
4.3	Mesure	33
5	Système à plusieurs qubits	35
5.1	Description du système	35
5.2	Les portes	36
5.3	Applications simples	38
5.3.1	Superdense Coding	38
5.3.2	Téléportation quantique	41
6	Un ordinateur quantique	44
6.1	Universalité	44
6.2	Hardware	45
III	Exemples d'algorithmes	49
7	Algorithme de Deutsch-Jozsa	50

8 Quelques protocoles	53
8.1 Retour de phase	53
8.2 Transformée de Fourier quantique	55
8.3 Estimation de phase quantique	57
9 Algorithme de Shor	61
9.1 Principe	61
9.2 Implémentation simple	62
9.2.1 Simulation	63
9.2.2 Hardware réel	64
9.3 Application à un problème concret	65
10 Cryptographie : distribution de clés	68
11 Algorithme de Grover	70
11.1 Principe	70
11.2 Comparaison avec une implémentation classique	72
11.2.1 2 entrées	73
11.2.2 3 entrées	73
11.2.3 4 entrées	74
11.2.4 Différence de complexité	76
11.3 Résolution d'un sudoku	76
12 Estimation de phase itérative	79
13 Modélisation d'un système physique	81
IV Et après...	85
14 Technologies de hardware	86
15 Sur des machines à court terme	88
16 Sur le long terme	89
17 Conclusion	90
Appendices	91
A Utilisation de <i>Qiskit</i>	92

Première partie

Préambule

Chapitre 1

Introduction

“C’est quoi un projet que tu trouves particulièrement intéressant que t’as vu ?” est la question posée à Octave Klaba ¹, président d’OVH, l’un des plus gros hébergeurs de sites web d’Europe. Il y répond “[...] Mais s’il y a un sujet, c’est vraiment quantique. En fait, il faut bien voir, c’est que tout le siècle dernier était basé sur l’atome [...] Celui-là, c’est le quantique.”

Cette citation est un bon exemple de l’importance de la physique quantique dans le domaine de l’informatique. En effet, parmi tous les domaines de recherches technologiques, comme l’amélioration des machines existantes, les recherches en informatique biologique avec par exemple le stockage de l’information dans l’ADN [1], ou encore les recherches en informatique quantique, c’est cette dernière qui est la plus mise en avant par un homme aussi bien placé dans le domaine de l’innovation technologique.

L’idée d’utiliser la physique quantique pour améliorer l’informatique est apparue dans les années 1980, proposée indépendamment par Richard Feynman, prix Nobel de physique en 1965, et Yuri Manin. Ce fut par la demande exponentielle de ressources informatiques, pour faire de la simulation de systèmes quantiques dynamiques, qu’émergea l’idée d’utiliser des ordinateurs basés sur des phénomènes quantiques.

Entre 1980 et 2000, le domaine de l’informatique et plus précisément de l’algorithmie quantique voit apparaître de nombreux résultats théoriques très prometteurs. Le premier cas d’un avantage théorique est montré par David Deutsch en 1985, via un cas de problème dit de “boîte noire”, puis ces problèmes vont être très étudié, comme la généralisation de celui étudié par Deutsch avec l’aide de Richard Jozsa.

Les idées d’applications concrètes apparaissent tout d’abord en 1984, avec la proposition d’utiliser les propriétés quantiques afin d’améliorer la sécurité des communications, via de potentiels nouveaux protocoles de cryptographie qui utiliseraient des clés dites “quantiques”.

D’un autre côté, en 1994, Peter Shor crée, en s’appuyant sur les résultats de ses prédécesseurs, un algorithme de factorisation des nombres entiers en facteurs premiers qui pourrait casser le système de sécurité alors très répandu *RSA*. Cela inquiéta suffisamment les spécialistes de la sécurité informatique pour que l’on crée des protocoles de cryptographie plus sûrs, et surtout résistant ces potentielles nouvelles technologies.

Les dernières avancées majeures de cette période sont dues à Lov Grover, proposant un algorithme quantique plus rapide que celui classique dans la résolution d’une catégorie de problèmes assez répandu. Dans cette même année 1996, Seth Lloyd prouve que les ordinateurs quantiques permettent de simuler des systèmes quantiques avec un gros avantage sur les simulations classiques [2].

En pratique, la première démonstration que la technologie est réalisable est la construc-

¹De **Underscore_** : *On a reçu le milliardaire qui fait trembler amazon*

tion d'un ordinateur quantique à deux qubits - unité en informatique quantique, équivalent au bit classique - en 1998. Au fur et à mesure, le nombre de qubits a augmenté et l'erreur devient de plus en plus réduite, à tel point que Google et la Nasa annoncent en 2019 qu'ils auraient atteint la suprématie quantique. Cette suprématie désigne le moment où un ordinateur quantique peut effectuer une tâche bien choisie en moins de temps qu'un ordinateur classique. Néanmoins, cette affirmation est très controversée et il est actuellement admis que l'affirmation était prématurée. Depuis cet ordinateur à 54 qubits, la recherche continue et aujourd'hui, en 2023, il existe certains ordinateurs à plus de 400 qubits, et l'objectif d'IBM, par exemple, est d'atteindre 4000 qubits d'ici à 2025 [3] (un processeur actuel possède des milliards de transistors - autour de 2 milliards - néanmoins cela correspondrait à une trentaine de qubits parfaits, mais ce n'est pas du tout réaliste, car en pratique, il faudrait peut-être plusieurs milliers de qubits réels pour un seul qubit logique [4, 5]).

On voit donc de nos jours que la recherche en informatique quantique est très active et que les plus grands acteurs de l'informatique, comme Google, IBM, Microsoft, Intel, ou encore la Nasa, sont très impliqués dans ce domaine. Cette recherche mobilise de nombreuses compétences, des ingénieurs qui créent les machines, des physiciens qui étudient les phénomènes quantiques, des mathématiciens qui développent les algorithmes, des informaticiens qui créent les logiciels, et des chercheurs qui étudient les applications possibles. Ces grandes entreprises cherchent aussi à attirer des jeunes dans ce domaine, en mettant à disposition des outils de simulation, des tutoriels, des formations, et en organisant des concours. L'accès à ces outils est aussi une manière pour les entreprises d'essayer de se garantir une place dans ce secteur si prometteur. Citons par exemple la création de langage de programmation spécifique comme le Q# [6] de Microsoft [7] ou le module Qiskit [8] d'IBM [9], en plus des solutions open sources existantes, ou la mise à disposition de simulateurs ou d'ordinateur quantique réel à des gens qui ne sont pas dans la recherche.

Ce travail vise à poser les bases des ordinateurs quantiques, ainsi que d'en présenter certains exemples qui montrent l'origine de l'intérêt pour cette technologie. Il faut malgré tout garder à l'esprit que cela demeure un domaine très jeune et qu'il demeure principalement théorique, et que les machines existantes actuellement sont encore soumis à des erreurs importantes.

Chapitre 2

Notions théoriques

Avant d'aborder le vif du sujet, il est nécessaire de définir quelques notions théoriques. En premier lieu, nous allons définir une méthode pour comparer les performances de différents algorithmes, car c'est un point central qui justifie la recherche en informatique quantique. Dans un second temps, nous allons définir les concepts de la mécanique quantique, sur lesquels s'appuient les algorithmes quantiques.

2.1 Informatique

La différence de technologie interroge sur la possibilité de comparer les performances de celles-ci. On voit bien que quelque chose est plus rapide qu'une autre en mesurant le temps, mais si la technologie est encore à bâtir, il demeure intéressant de comparer les différents algorithmes. On peut se dire qu'en les exécutant sur un ordinateur, on peut comparer les temps d'exécution. Mais cela n'est pas fixe et reproductible, car cela dépend de tout un tas de facteurs.

Pour illustrer cela, comparons sur une même machine le même algorithme, mais avec deux langages différents. Premièrement, on va utiliser le Python, et deuxièmement, on va utiliser le C. Le principe de l'algorithme est de calculer la factorielle d'un nombre.

$$n! = \prod_{i=1}^n i = 1 \times 2 \times 3 \times \dots \times (n-1) \times n, n \in \mathbb{N}$$

Pour cela, on va utiliser l'algorithme suivant :

Algorithme 1 : Factorielle

Entrées : Un nombre entier n
Sorties : La factorielle de n : $n! = r$
 $N \leftarrow n$;
 $r \leftarrow 1$;
 $i \leftarrow 1$;
tant que $i \leq N$ **faire**
 $r \leftarrow r \times i$;
 $i \leftarrow i + 1$;
fin
retourner r ;

Ce qui s'implémente comme suit dans les différents langages :

Factorielle en Python

```

1 def factorial(n):
2
3     result = 1
4     for i in range(1, n+1):
5         result *= i
6
7     return result

```

Factorielle en C

```

1 int factorial(int n) {
2
3     int result = 1;
4     for (int i = 1; i <= n; i++)
5     {
6         result *= i;
7     }
8     return result;
9 }

```

Afin de comparer les temps d'exécution, on va appeler la fonction 10000 fois, et on va calculer la moyenne.

Programme de test en Python

```

1 import timeit
2
3 p = 20
4 def factorial(n):
5     result = 1
6     for i in range(1, n+1):
7         result *= i
8
9     return result
10
11
12 f = factorial(p)
13 elapsed_time = timeit.timeit(str((factorial(p))), number=10000)
14
15 print("Factorial of", p, "=", factorial(p))
16 print("Time measured :", elapsed_time, "seconds.")
17
18
19 out :
20 Factorial of 20 = 2432902008176640000
21 Time measured : 8.160003926604986e-05 seconds.

```

Dans ce premier programme, on peut noter utilisation du package `timeit` qui permet de mesurer le temps d'exécution. De plus, le temps d'exécution est très court, de l'ordre de 10^{-5} secondes.

Programme de test en C

```

1 #include <stdio.h>
2 #include <sys/time.h>
3
4 unsigned long long int factorial(int n)
5 {
6     unsigned long long int result = 1;
7     for (int i = 1; i <= n; i++)
8     {
9         result *= i;

```

```
10     }
11     return result;
12 }
13
14 int main()
15 {
16     int n = 20;
17
18     struct timeval begin, end;
19     gettimeofday(&begin, 0);
20
21     for (int i = 0; i < 10000; i++)
22     {
23         factorial(n);
24     }
25
26     gettimeofday(&end, 0);
27     long seconds = end.tv_sec - begin.tv_sec;
28     long microseconds = end.tv_usec - begin.tv_usec;
29     double elapsed = (seconds + microseconds*1e-6)/10000;
30
31     printf("Factorial of %d = %llu\n", n, factorial(n));
32     printf("Time measured: %.9f seconds.\n", elapsed);
33
34     return 0;
35 }
36
37
38 out :
39 Factorial of 20 = 2432902008176640000
40 Time measured: 0.000000019 seconds.
41
42 Process finished with exit code 0
```

Dans ce second programme, la mesure du temps est plus compliquée, car il faut utiliser des fonctions spécifiques à la librairie `sys/time.h`, sans disposer d'outils dédiés. De plus, la structure du C est plus complexe que celle du Python. Néanmoins, le temps d'exécution est toujours très court, de l'ordre de 10^{-8} secondes, soit plus de 1000 fois plus rapide.

De plus, on peut faire une constatation similaire sur le temps d'exécution de la fonction selon la taille de l'entrée.

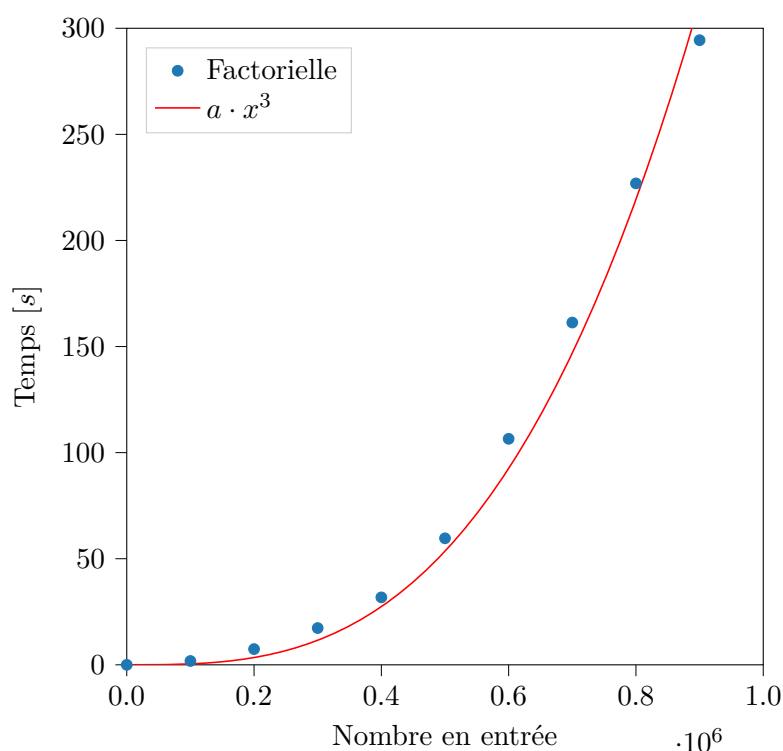


FIGURE 2.1 : Temps d'exécution de la fonction factorielle en Python selon la taille de l'entrée

En premier abord, on peut penser que le temps d'exécution est proportionnel à la taille de l'entrée. En effet, on fait une boucle de 1 à n , et on fait une multiplication à chaque itération. Or, on constate que le temps d'exécution est plus proche d'une fonction cubique que linéaire. Cela est dû à la multiplication qui est plus longue pour un grand nombre.

On constate donc que sur une même machine, deux programmes donnent des temps très différents pour un même algorithme. À noter malgré tout que le C est un langage compilé alors que le Python est un langage interprété, ce qui explique en partie la différence de temps d'exécution, mais pose malgré tout un problème de comparaison. De plus, leurs temps d'exécution varient selon la taille de l'entrée. Dès lors, comment comparer les temps d'exécution de deux algorithmes différents ?

Les algorithmes sont alors classés selon leur complexité [10], soit leur “temps d'exécution” selon la taille de l'entrée, indépendamment de la machine utilisée. Ce n'est néanmoins pas la définition exacte de la complexité, car c'est plutôt l'ordre de grandeur du nombre d'opérations à effectuer qui est important, mais cela donne une excellente base de comparaison pour de grandes valeurs. Symboliquement, on utilise la notation $\mathcal{O}()$ [11] pour exprimer la complexité d'un algorithme (à noter que d'autres notations existent pour des usages similaires, comme $\Theta()$ ou $\Omega()$).

La définition de la complexité que l'on va utiliser est la suivante :

$$f(x) = \mathcal{O}(g(x)) \text{ quand } x \rightarrow \infty$$

si la valeur absolue de $f(x)$ est à un multiple constant de $g(x)$ près, pour des valeurs de x suffisamment grandes. Pour simplifier l'écriture, on écrira juste $f(x) = \mathcal{O}(g(x))$.

Cela permet d'expliquer la croissance de la factorielle, car on estime que la multiplication est en $\mathcal{O}(n^2)$ (pour simplifier les calculs, mais sur les grands nombres, c'est un peu

moins [12]), donc en la faisant n fois, on obtient une complexité de $\mathcal{O}(n^3)$.

Prenons quelques autres exemples d'algorithmes afin de mieux comprendre ce concept :

1. Calculer $(-1)^n$: le temps ne dépend pas de l'entrée donc la complexité est $\mathcal{O}(1)$

Algorithme 2 : Calcul de $(-1)^n$

Entrées : Un nombre entier n
Sorties : Le résultat de $(-1)^n$: r
 $N \leftarrow n$;
si N est pair **alors**
 | $r \leftarrow 1$;
sinon
 | $r \leftarrow -1$;
fin
retourner r ;

2. Trouver le maximum d'une liste de nombres : le temps dépend uniquement de la taille de la liste donc la complexité est $\mathcal{O}(n)$

Algorithme 3 : Recherche du maximum d'une liste

Entrées : Une liste de nombres L
Sorties : Le maximum de la liste : m
 $m \leftarrow L[0]$;
pour $i \leftarrow 1$ **à** n **faire**
 | **si** $L[i] > m$ **alors**
 | $m \leftarrow L[i]$;
 | **fin**
fin
retourner m ;

3. Calculer naïvement le produit matriciel de deux matrices carrées de taille n : il faudra calculer n -fois la somme des n -lignes multipliée par les n -colonnes, donc $n \times n \times n = n^3$, soit $\mathcal{O}(n^3)$

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \times \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mp} \end{pmatrix}$$

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj} = \sum_{k=1}^n a_{ik}b_{kj}$$

Produit matriciel d'une matrice A de taille $m \times n$ et d'une matrice B de taille $n \times p$

Algorithme 4 : Produit matriciel naïf**Entrées :** Deux matrices carrées A et B de taille n **Sorties :** Le produit matriciel C de A et B

```

pour  $i \leftarrow 1$  à  $n$  faire
    pour  $j \leftarrow 1$  à  $n$  faire
         $C[i, j] \leftarrow 0$  ;
        pour  $k \leftarrow 1$  à  $n$  faire
             $C[i, j] \leftarrow C[i, j] + A[i, k] \times B[k, j]$  ;
        fin
    fin
fin
retourner  $C$ ;
```

Sur le dernier exemple, le terme naïf est utilisé, car il existe des algorithmes plus efficaces pour calculer le produit matriciel, comme par exemple l'algorithme de Strassen [13], qui a une complexité de $\mathcal{O}(n^{2.807})$, ou celui de Coppersmith-Winograd avec $\mathcal{O}(n^{2.376})$ [14] (Notons qu'il faudrait pour uniquement lire les entrées des matrices une complexité $\mathcal{O}(n^2)$).

Pour illustrer cette différence de complexité entre deux algorithmes, on va utiliser la recherche d'un nombre dans une liste ordonnée. Pour rechercher un nombre dans une liste, l'algorithme le plus simple est de parcourir la liste jusqu'à trouver le nombre, ou jusqu'à la fin de la liste, avec une complexité de $\mathcal{O}(n)$ (voir Algorithme 3).

Mais si la liste est ordonnée, on peut optimiser la recherche selon le principe de recherche dichotomique. On va comparer le nombre recherché avec le nombre au milieu de la liste, et si le nombre recherché est plus grand, on va refaire la même opération sur la moitié supérieure de la liste, sinon sur la moitié inférieure.

Algorithme 5 : Recherche dichotomique**Entrées :** Liste L ordonnée, nombre n **Sorties :** Position de n dans L , ou -1 si n n'est pas dans L

```

 $i \leftarrow 0$ ;
 $j \leftarrow |L|$ ;
tant que  $i < j$  faire
     $m \leftarrow \lfloor \frac{i+j}{2} \rfloor$ ;
    si  $L[m] < n$  alors
         $i \leftarrow m + 1$ ;
    fin
    sinon si  $L[m] > n$  alors
         $j \leftarrow m$ ;
    fin
    sinon
        retourner  $m$ ;
    fin
fin
retourner  $-1$ ;
```

On voit donc que l'on divise notre liste par deux à chaque itération. Si on considère que la taille de la liste est n et x le nombre d'opérations nécessaires pour trouver le nombre, on a donc $n \times 2^{-x} = 1$, soit $x = \log_2(n)$, d'où une complexité de $\mathcal{O}(\log(n))$ (pour la complexité, la base n'est pas précisée, car par la formule du changement de base du logarithme, l'un ne varie de l'autre que par un scalaire).

On peut donc voir que la recherche dichotomique est beaucoup plus efficace que la recherche naïve, et que lorsque la taille de la liste augmente, la différence de temps d'exécution entre les deux algorithmes augmente de manière exponentielle.

Ce genre d'amélioration est très important en informatique, car il permet de résoudre des problèmes qui seraient autrement insolubles. Ces optimisations sont presque toutes aussi importantes qu'une bonne implémentation, et il est donc utile de les connaître. La complexité des algorithmes quantiques vis-à-vis des algorithmes classiques est un des points centraux justifiant la recherche dans ce domaine, parce que cela permettrait des accélérations considérables dans de nombreux domaines.

Il faut quand même prendre en compte que lorsque l'on implémente un algorithme, la complexité réelle est souvent plus grande que la complexité théorique, car il faut prendre en compte les opérations de base, comme les additions, les multiplications, les comparaisons, etc. Ces opérations sont souvent considérées comme ayant une complexité $\mathcal{O}(1)$, mais en réalité, elles dépendent de la taille des nombres manipulés, et donc de la taille de l'entrée.

Si on reprend l'Algorithme 2, pour faire $(-1)^n$, on regarde la parité de n , or pour cela, il y a plusieurs façons de le réaliser. Premièrement, on peut regarder le dernier bit de n , et si il est à 1, alors n est impair, sinon il est pair, ou en base 10 si le dernier chiffre est dans $\{0,2,4,6,8\}$, alors c'est pair et sinon impair. Ceci est une opération en $\mathcal{O}(1)$, car on ne fait qu'une comparaison. Mais si on n'y pense pas, on peut également faire $n \bmod 2$ (donc si le résultat est 0, c'est pair et sinon impair), qui est une opération beaucoup plus coûteuse, parce qu'elle nécessite une division.

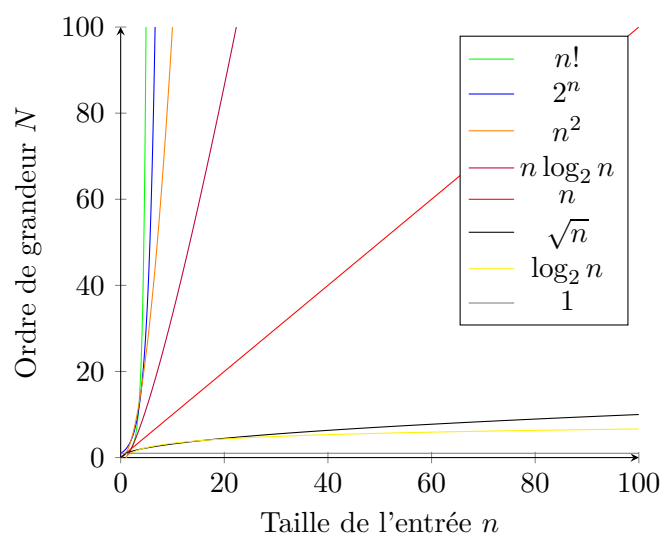


FIGURE 2.2 : Comparaison de quelques complexités courantes

Sur le graphe ci-dessus, on peut voir clairement l'enjeu d'optimiser la complexité d'un algorithme, d'autant plus que la taille de l'entrée augmente. Cela permet de surpasser des obstacles qui seraient potentiellement insurmontables avec des améliorations de performances matérielles uniquement, sur lesquelles on compte principalement dans notre société actuelle en vertu de la loi de Moore [15].

Cette question de la complexité des algorithmes peut être étendue à la théorie de la complexité, visant à étudier la difficulté des problèmes, et à les classer selon leur complexité et également étudier les relations entre les classes de complexité [16]. Par exemple, on va dénommer P la classe des problèmes qui peuvent être résolus en temps polynomial, où nous

pouvons citer le problème de la multiplication de deux matrices. Une autre classe célèbre est NP , qui contient les problèmes qui peuvent être vérifiés en temps polynomial, comme par exemple un sudoku : il est facile de vérifier si une grille est correcte, mais il est difficile de la résoudre de manière générale.

Cette question des classes de complexité est très importante, à tel point qu'elle est considérée comme l'un des sept problèmes du prix du millénaire, celui nommé vulgairement $P = NP$ et visant à déterminer si tous les problèmes vérifiables en temps polynomial sont également résolubles en temps polynomial. Ce sont sept problèmes mathématiques qui ont été définis par le Clay Mathematics Institute en 2000, et pour lesquels une récompense d'un million de dollars est promise à celui qui les résoudra. Les algorithmes quantiques ne remettent pas directement en cause cette question, mais ils en ouvrent de nouvelles, dans ce domaine, car ils permettent de résoudre des problèmes qui sont dans NP en temps polynomial avec une probabilité assez élevée [17].

En résumé, la complexité d'un algorithme est une mesure de sa difficulté, et permet de comparer les performances de différents algorithmes indépendamment de la machine utilisée. Cette mesure peut être tout autant dans le temps d'exécution que dans l'espace mémoire utilisé. Néanmoins, de par sa définition et sa nature asymptotique, la mesure de la complexité est vraiment pertinente quand les entrées sont grandes, car sinon il faut prendre en compte les machines utilisées.

2.2 Physique

Derrière le terme d'algorithme quantique se trouve les concepts de la mécanique quantique. Celle-ci vise à décrire le comportement de la matière à l'échelle atomique et subatomique. Elle est basée sur le principe de dualité onde-corpuscule, qui dit que selon l'expérience envisagée une particule (objet localisé dans l'espace) se comporte comme une onde (phénomène étendu) et inversement une onde se comporte comme une particule.

Plus concrètement, cette idée peut déjà se pressentir avec les études sur la lumière dès le 19^{ème} siècle. En effet, c'est en 1801 que Thomas Young réalise l'expérience qui porte son nom, les fentes de Young [18], qui met en évidence l'aspect ondulatoire de la lumière par les franges d'interférence qui résultent de la superposition des ondes émises par chacun des deux trous. Avec des particules classiques on verrait uniquement deux taches suivant que la particule est passée par l'un ou l'autre trou .

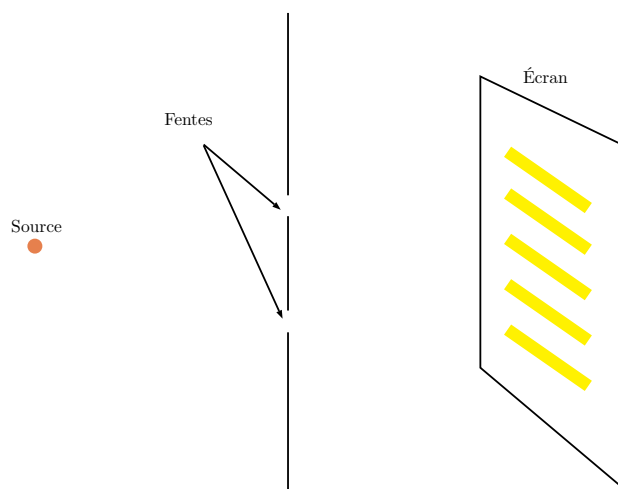


FIGURE 2.3 : Schéma de l'expérience des fentes de Young

Le principe de cette expérience est de faire passer un faisceau de lumière à travers deux fentes parallèles, et d'observer le résultat sur un écran. Là où l'on s'attendrait à voir deux taches lumineuses, on observe en réalité une multitude de taches, propres aux franges d'interférences entre deux ondes. Cette conception de la lumière comme une onde semble être alors la plus probable, car la théorie de Maxwell, qui décrit les ondes électromagnétiques, permet de décrire la lumière comme tel.

C'est en 1900 que l'idée de quantification va émerger [19] avec les travaux de Max Planck sur le problème du rayonnement du corps noir. Il obtient de manière empirique la formule pour ce rayonnement. Pour interpréter cette équation, il introduit l'hypothèse que la matière ne peut rayonner de l'énergie que par des quantités finies - ou quanta - proportionnelles à la fréquence de vibration des atomes. Toutefois, il n'envisage pas que la lumière puisse avoir une structure discontinue. Au contraire pour élaborer sa théorie de l'effet photoélectrique (découvert en 1887) [20] Einstein introduit en 1905 l'hypothèse que lors de l'émission ou l'absorption d'un rayonnement lumineux par la matière, celui-ci doit être considéré comme formé d'un nombre fini de petits grains indivisibles, ou "quanta", parfaitement localisés dans l'espace, dont l'énergie E et la quantité de mouvement p sont proportionnelles à la fréquence de l'onde, soit

$$E = h\nu \quad \text{et} \quad p = h\frac{\nu}{c}$$

où h est la constante de Planck, ν la fréquence de l'onde et c la vitesse de la lumière.

Au contraire lors de sa propagation la lumière doit être considérée comme une onde obéissant aux équations de Maxwell. Mais les physiciens ont du mal à accepter cette hypothèse, en particulier Max Planck qui dans un rapport élogieux sur Einstein écrit en 1913 "il est vrai qu'il a parfois manqué le but lors de ses spéculations par exemple avec son hypothèse sur les quanta lumineux."

L'hypothèse d'Einstein sera confirmée en 1923 par les expériences de Compton et en 1926 on donnera le nom de photon à cette particule lumineuse.

En 1923 le physicien français Louis de Broglie suggère d'associer à toute particule massive d'énergie E et de quantité de mouvement $p = mv$ une onde de fréquence ν et de longueur d'onde λ avec

$$\nu = \frac{E}{h} \quad \text{et} \quad \lambda = \frac{h}{p}$$

Cette hypothèse lui permet d'expliquer les règles de quantification des orbites électroniques dans l'atome introduite par Bohr en 1913 par décrire les spectres atomiques [21].

De ces théories, le physicien allemand Erwin Schrödinger va se poser la question de connaître l'équation d'onde qui décrit le comportement d'une particule. Dans ce qui suit, on va décrire un chemin afin de la comprendre avec une dimension spatiale et une dimension temporelle, et la généralisation va être mentionnée sans démonstration.

Mais avant de commencer, il faut définir ce qu'est une onde. Une onde est une perturbation qui se propage dans un milieu, et qui transporte de l'énergie sans transporter de matière. La perturbation peut être de différentes natures, comme une perturbation mécanique, électromagnétique, etc. On peut la décrire de manière classique par une fonction $y(x, t)$, où x est la position dans l'espace et t est le temps. Cette fonction est solution de l'équation d'onde classique à une dimension :

$$\frac{\partial^2 y}{\partial x^2} = \frac{1}{v^2} \frac{\partial^2 y}{\partial t^2}$$

où v est la vitesse de propagation de l'onde.

Démonstration. Pour montrer que cette équation fonctionne bel et bien, on va considérer comme onde une corde vibrante en deux dimensions [22]. Prenons un tronçon de corde infinitésimal AB avec $A(x; y)$ et $B(x + dx; y + dy)$, et l'angle entre la tangente à la corde et l'horizontal en ce point au temps t donné par $\alpha(x, t)$ (respectivement $\alpha(x + dx, t)$). En négligeant la force pesante, on a uniquement la force de tension qui agit sur la corde, \vec{T}_A en A et \vec{T}_B en B .

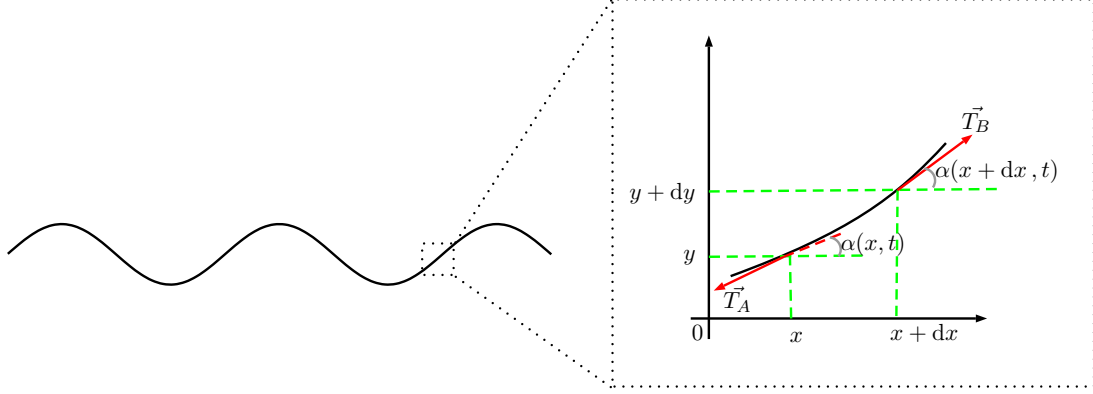


FIGURE 2.4 : Onde sur une corde et forces considérées

Connaissant la masse linéique de la corde μ , on peut écrire la relation suivante :

$$\vec{T}_A + \vec{T}_B = \mu dx \vec{a}$$

où $\vec{a} = \begin{pmatrix} \frac{\partial^2 x}{\partial t^2} \\ \frac{\partial^2 y}{\partial t^2} \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{\partial^2 y}{\partial t^2} \end{pmatrix}$ est l'accélération de la corde (nulle horizontalement, car il n'y a qu'un déplacement vertical).

On peut décomposer l'équation en deux équations, une selon l'axe horizontal Ox et une selon l'axe verticale Oy .

$$\begin{cases} Ox : -T_A \cos \alpha(x, t) + T_B \cos \alpha(x + dx, t) = 0 \\ Oy : -T_A \sin \alpha(x, t) + T_B \sin \alpha(x + dx, t) = \mu dx \frac{\partial^2 y}{\partial t^2} \end{cases}$$

En utilisant les approximations du premier ordre $\sin \alpha \approx \alpha$ et $\cos \alpha \approx 1$, si l'on considère la variation créée par l'onde et donc les angles comme faibles, on obtient :

$$\begin{cases} Ox : -T_A + T_B = 0 \\ Oy : -T_A \alpha(x, t) + T_B \alpha(x + dx, t) = \mu dx \frac{\partial^2 y}{\partial t^2} \end{cases} \Rightarrow \begin{cases} T_A = T_B = T \\ -T \alpha(x, t) + T \alpha(x + dx, t) = \mu dx \frac{\partial^2 y}{\partial t^2} \end{cases}$$

De plus, nous pouvons considérer que $\alpha = \tan \alpha = \frac{\partial y}{\partial x}$, d'où :

$$-T \frac{\partial y}{\partial x}(x, t) + T \frac{\partial y}{\partial x}(x + dx, t) = \mu dx \frac{\partial^2 y}{\partial t^2}(x, t)$$

En notant que

$$\frac{\partial y}{\partial x}(x + dx, t) - \frac{\partial y}{\partial x}(x, t) = \frac{\partial}{\partial x}(y(x + dx, t) - y(x, t)) = \frac{\partial}{\partial x} \left(\frac{\partial y}{\partial x}(x, t) dx \right) = \frac{\partial^2 y}{\partial x^2} dx$$

on obtient

$$T \frac{\partial^2 y}{\partial x^2} dx = \mu dx \frac{\partial^2 y}{\partial t^2} \Leftrightarrow \frac{\partial^2 y}{\partial x^2} = \frac{\mu}{T} \frac{\partial^2 y}{\partial t^2}$$

qui est une équation d'onde, ou équation de d'Alembert, de forme

$$\frac{\partial^2 y}{\partial x^2} = \frac{1}{v^2} \frac{\partial^2 y}{\partial t^2}$$

où $v = \sqrt{\frac{T}{\mu}}$ est la célérité de l'onde sur la corde dans ce cas. \square

On peut noter que la célérité est reliée à la fréquence ν et à la longueur d'onde λ par la relation $v = \lambda\nu$. De plus, on peut remarquer que l'équation d'onde est une équation linéaire, c'est-à-dire que si y_1 et y_2 sont deux solutions de l'équation, alors $y = y_1 + y_2$ est aussi une solution, ce qui va induire le principe dit de superposition.

Sachant cela, essayons de construire une équation d'onde pour une particule [23]. Dans le cas classique, une solution possible est une onde plane, de la forme

$$y(x, t) = A \cos(kx - \omega t)$$

Avec $k = \frac{2\pi}{\lambda}$ le nombre d'onde et $\omega = 2\pi\nu$ la pulsation. Or, des formules $E = h\nu$ et $p = \frac{h}{\lambda}$, on peut déduire que tant qu'il n'y a pas de forces extérieures, la quantité de mouvement p ne varie pas et donc la longueur d'onde λ non plus. Mais pour pouvoir construire une équation générale, il faut savoir comment cela se passe aussi dans le cas où cela n'est pas vrai. L'équation doit donc correspondre à plusieurs postulats :

1. Les postulats de De Broglie-Einstein : $E = h\nu$ et $p = \frac{h}{\lambda}$
2. L'équation d'énergie $E = K + P$ avec E l'énergie totale, K l'énergie cinétique et P l'énergie potentielle ; de plus, $K = \frac{1}{2}mv^2 = \frac{p^2}{2m}$ (avec m la masse de la particule, v sa vitesse et p sa quantité de mouvement) et $P = V(x, t)$ avec V le potentiel d'où $F = -\frac{\partial V}{\partial x}$; ainsi, $E = \frac{p^2}{2m} + V$
3. L'équation doit être linéaire (comme l'équation d'onde)
4. Pour un potentiel V constant, l'équation différentielle doit avoir une onde sinusoïdale comme solution de longueur d'onde et de fréquence constantes (car $V(x, t) = \text{constante} \Rightarrow F = -\frac{\partial V}{\partial x} = 0$ et donc $p = \text{constante}$ car $F = \frac{dp}{dt}$)

De ces postulats, on déduit que

$$E = \frac{p^2}{2m} + V(x, t) = \frac{h^2}{2m\lambda^2} + V(x, t) = h\nu$$

puis en remplaçant ν par $\frac{\omega}{2\pi}$ et λ par $\frac{2\pi}{k}$, ainsi qu'utilisant la constante de Planck réduite $\hbar = \frac{h}{2\pi}$ afin de simplifier la notation, on obtient

$$\frac{\hbar^2 k^2}{2m} + V(x, t) = \hbar\omega$$

sur laquelle on note un facteur k^2 à gauche et un facteur ω à droite. De la 4^e condition, commençons par considérer que la fonction recherchée est de la forme

$$\Psi(x, t) = \cos(kx - \omega t)$$

et de là notons que

$$\frac{\partial^2 \Psi(x, t)}{\partial x^2} = -k^2 \Psi(x, t) \quad \text{et} \quad \frac{\partial \Psi(x, t)}{\partial t} = \omega \sin(kx - \omega t)$$

d'où la conclusion que l'équation doit contenir une dérivée seconde par rapport à x et une dérivée première par rapport à t , ainsi qu'un facteur avec le potentiel $V(x, t)$, multiplié par $\Psi(x, t)$ afin d'assurer la linéarité de l'équation.

Posons donc une équation de la forme

$$\alpha \frac{\partial^2 \Psi(x, t)}{\partial x^2} + V(x, t) \Psi(x, t) = \beta \frac{\partial \Psi(x, t)}{\partial t}$$

où α et β sont des constantes à déterminer.

En essayant de vérifier la condition 4, on pose $\Psi(x, t) = \cos(kx - \omega t)$ et $V(x, t) = V_0$, et on regarde ce que cela donne :

$$\begin{aligned} \alpha \frac{\partial^2 \Psi(x, t)}{\partial x^2} + V_0 \Psi(x, t) &= \beta \frac{\partial \Psi(x, t)}{\partial t} \\ \Leftrightarrow -\alpha k^2 \cos(kx - \omega t) + V_0 \cos(kx - \omega t) &= \beta \omega \sin(kx - \omega t) \\ \Leftrightarrow (\alpha k^2 - V_0) \cos(kx - \omega t) + \beta \omega \sin(kx - \omega t) &= 0 \end{aligned}$$

Cette égalité ne se vérifie que si $k^2 = \frac{V_0}{\alpha}$ et $\beta = 0$, or sous ces conditions, on ne satisfait pas l'équation d'énergie $\frac{\hbar^2 k^2}{2m} + V(x, t) = \hbar \omega$.

Nous devons donc considérer une autre forme pour $\Psi(x, t)$, et nous allons essayer avec

$$\Psi(x, t) = \cos(kx - \omega t) + \gamma \sin(kx - \omega t)$$

où γ est une constante à déterminer. On a alors

$$\begin{aligned} \frac{\partial \Psi(x, t)}{\partial t} &= \omega (\sin(kx - \omega t) - \gamma \cos(kx - \omega t)) \\ \frac{\partial^2 \Psi(x, t)}{\partial x^2} &= -k^2 \cos(kx - \omega t) - \gamma k^2 \sin(kx - \omega t) \end{aligned}$$

et donc

$$\begin{aligned} \alpha \frac{\partial^2 \Psi(x, t)}{\partial x^2} + V(x, t) \Psi(x, t) &= \beta \frac{\partial \Psi(x, t)}{\partial t} \\ \Leftrightarrow -\alpha k^2 \cos(kx - \omega t) - \alpha \gamma k^2 \sin(kx - \omega t) + V_0 (\cos(kx - \omega t) + \gamma \sin(kx - \omega t)) &= \\ &= \beta \omega (\sin(kx - \omega t) - \gamma \cos(kx - \omega t)) \\ \Leftrightarrow -\alpha k^2 [\cos(kx - \omega t) + \gamma \sin(kx - \omega t)] + V_0 [\cos(kx - \omega t) + \gamma \sin(kx - \omega t)] &= \\ &= \frac{\beta \omega}{\gamma} [-\gamma^2 \cos(kx - \omega t) + \gamma \sin(kx - \omega t)] \end{aligned}$$

qui contient dans les crochets à gauche du signe égal la fonction $\Psi(x, t)$, et dans les crochets à droite du signe égal une fonction qui y ressemble, mais qui n'est pas exactement la même.

Nous pourrions alors essayer de trouver une valeur de γ qui permet de faire correspondre les deux fonctions, soit satisfaisant l'équation

$$-\gamma^2 \cos(kx - \omega t) + \gamma \sin(kx - \omega t) = \cos(kx - \omega t) + \gamma \sin(kx - \omega t)$$

ce qui nécessite que $\gamma^2 = -1$, donc $\gamma = \pm \sqrt{-1} = \pm i$, où i est l'unité imaginaire. En posant $\gamma = i$, on peut simplifier l'équation précédente en

$$-\alpha k^2 + V_0 = \frac{\beta \omega}{i}$$

et en comparant avec l'équation de De Broglie-Einstein $\frac{\hbar^2 k^2}{2m} + V(x, t) = \hbar\omega$, on voit que l'équation précédente est satisfaite si $\alpha = -\frac{\hbar^2}{2m}$ et $\beta = i\hbar$.

Nous avons donc trouvé une équation qui satisfait les conditions 1, 2, 3 et 4, et qui est donc une équation d'onde quantique pour une particule de masse m dans un potentiel $V(x, t)$, à savoir

$$-\frac{\hbar^2}{2m} \frac{\partial^2 \Psi(x, t)}{\partial x^2} + V(x, t) \Psi(x, t) = i\hbar \frac{\partial \Psi(x, t)}{\partial t}$$

dénommée équation de Schrödinger (à noter que tout comme il existe une équation d'onde de dimension plus grande, il y a aussi une équation de Schrödinger de dimension supérieure).

On admet que pour un instant t_0 donné la fonction d'onde $\Psi(x, t_0)$ décrit l'état de la particule à cet instant. Comme l'équation de Schrödinger est linéaire, on remarque que connaissant deux états, définis par deux solutions de cette équation, on obtient par superposition linéaire de ces deux solutions une nouvelle solution, c'est-à-dire un nouvel état. C'est ce que l'on appelle "principe de superposition des états" : à partir de deux états de la particule, on peut définir de nouveaux états par combinaisons linéaires. C'est un principe entièrement nouveau qui n'a pas de correspondance en physique classique. En utilisant la notation introduite par Dirac, on représente cet état sous la forme d'un vecteur $|\Psi_t\rangle$, appelé "ket".

Cette équation prédit excellemment bien les résultats expérimentaux de la mécanique quantique, mais son interprétation est plus difficile, à commencer par que représente $\Psi(x, t)$, pourquoi l'unité imaginaire est-elle présente dans l'équation, etc.

Cela peut s'expliquer par la différence majeure entre la mécanique classique et la mécanique quantique : en mécanique classique, la mesure d'une grandeur physique ne change pas la valeur de n'importe quelle autre grandeur physique, alors qu'en mécanique quantique, la mesure d'une grandeur physique change la valeur de certaines autres grandeurs physiques. Cela s'exprime par les relations d'incertitude d'Heisenberg, qui sont des relations entre les incertitudes de deux grandeurs physiques, et qui sont données par

$$\Delta x \Delta p_x \geq \frac{\hbar}{2} \quad \text{et} \quad \Delta E \Delta t \geq \frac{\hbar}{2}$$

qui implique que plus on connaît précisément la position d'une particule, moins on connaît précisément son impulsion, et vice-versa, et plus on connaît précisément l'énergie d'une particule, moins on connaît précisément le temps qu'elle mettra à parcourir une certaine distance, et réciproquement.

L'équation de Schrödinger est donc une équation qui décrit l'évolution de la fonction d'onde d'une particule dans le temps, mais comme elle est complexe, elle ne peut pas être interprétée comme une mesure directe. Or, on sait que le module carré d'un nombre complexe est un nombre réel positif, et donc on peut interpréter cela comme une mesure. L'interprétation de la fonction d'onde donnée par Max Born est que son module au carré est la probabilité de trouver la particule dans un certain état, et donc l'équation de Schrödinger est une équation qui décrit l'évolution de la probabilité de trouver une particule dans un certain état dans le temps.

Dans le cas de la position, qui est une variable continue, la probabilité de trouver la particule dans l'intervalle $[x_1, x_2]$ est donnée par

$$P(x_1 \leq x \leq x_2) = \int_{x_1}^{x_2} |\Psi(x, t)|^2 dx$$

ce qui nous indique également, comme la somme des probabilités doit être égale à 1, que la fonction d'onde doit être normalisée, c'est-à-dire que

$$\int_{-\infty}^{+\infty} |\Psi(x, t)|^2 dx = 1$$

Cependant, dans la suite, on va considérer des grandeurs qui ne prennent que des valeurs discrètes, car l'informatique quantique fonctionne à l'image de celle classique, c'est-à-dire avec des bits, qui ne peuvent prendre que deux valeurs, 0 ou 1. De plus on va utiliser la notation de Dirac afin de représenter les états quantiques, qui est une notation qui permet de représenter les états quantiques de manière plus simple et plus compacte.

On va donner un exemple de cette notation, et on en posera les concepts plus rigoureusement par la suite. La grandeur spin a été mise en évidence en 1922 par O. Stern et W. Gerlach qui voulaient mesurer le moment angulaire d'un atome pour tester l'hypothèse de quantification du plan des orbites des électrons introduite par Sommerfeld. Le moment angulaire étant parallèle au moment magnétique, ils envoient des atomes à travers un champ magnétique et ils s'attendent à trouver sur un écran, situé après l'aimant, un nombre impair de taches correspondant aux différentes directions possibles du moment angulaire. Ils observent deux taches ce qui est un résultat incompréhensible. L'explication sera donnée en 1925 par G. Uhlenbeck et S. Goudsmit qui introduisent l'idée véritablement révolutionnaire que l'électron possède un moment angulaire intrinsèque, grandeur vectorielle sans équivalent classique, qui ne peut prendre deux valeurs $+\frac{1}{2}$ ou $-\frac{1}{2}$ quand mesurée dans une direction donnée. Pauli donnera le nom de spin à cette nouvelle grandeur. Comme elle ne peut prendre que deux valeurs, on peut représenter ces deux états par deux vecteurs, qui sont appelés vecteurs d'état, par exemple $|+\rangle$ et $|-\rangle$ pour les deux états de spin.

On représente alors l'état de spin de la particule par un vecteur qui est une combinaison linéaire des deux vecteurs d'état, par exemple $|\psi\rangle = c_1|+\rangle + c_2|-\rangle$, comme les deux états sont "solutions" de notre problème, et donc on peut représenter l'état de la particule par un vecteur qui est une combinaison linéaire de ceux-ci en vertu du 3^e postulat évoqué précédemment. Selon l'interprétation de Born, la probabilité de trouver la particule dans l'état $|+\rangle$ est donnée par $|c_1|^2$, et la probabilité de trouver la particule dans l'état $|-\rangle$ est donnée par $|c_2|^2$, et donc la somme des probabilités est égale à 1, ce qui implique que $|c_1|^2 + |c_2|^2 = 1$.

Ce formalisme peut paraître étrange, car il affirme que la particule n'est ni dans l'état $|+\rangle$ ni dans l'état $|-\rangle$, mais que si on fait une mesure, on l'observera soit dans l'état $|+\rangle$, soit dans l'état $|-\rangle$ avec un résultat aléatoire dont la probabilité est connue et après la mesure la particule sera dans l'état observé avec probabilité 1. Or ce phénomène est réel, et il est appelé superposition quantique [24]. C'est similaire au comportement lorsque deux ondes classiques se rencontrent, elles vont en quelque sorte s'additionner, comme à l'image de deux vagues qui se rencontrent en créant des vaguelettes l'une sur l'autre par endroit, augmentant leur taille à d'autre ou encore s'annulant en certains points.

Le fameux chat de Schrödinger [25] est une expérience de pensée visant à illustrer ce phénomène. Un dispositif diabolique est mis en place, qui consiste en une boîte hermétique contenant un chat, un flacon de poison, un détecteur de radioactivité et une source radioactive, et si le détecteur détecte une particule radioactive, alors le flacon de poison est cassé, et le chat meurt. Or le fait que la particule radioactive se désintègre ou non est un phénomène quantique, et donc tant que l'on ne regarde pas dans la boîte, on ne sait pas si la particule s'est désintégrée ou non, et donc si le chat est mort ou vivant. Cela paraît impossible et montre les lacunes de l'interprétation de la mécanique quantique, mais des phénomènes semblables peuvent être réalisés en laboratoire.

Citons par exemple l'interféromètre de Mach-Zehnder [26], qui est un dispositif permettant de mettre en évidence le phénomène d'interférence quantique, qui est un phénomène qui n'a pas d'équivalent en mécanique classique. C'est un dispositif qui consiste en un laser, qui est une source de photons, qui sont des particules de lumière, qui est envoyé sur un miroir semi-réfléchissant, qui va diviser le faisceau en deux faisceaux, qui vont suivre des chemins différents, et qui vont être réfléchis par deux miroirs, et sont redirigés vers un autre miroir

semi-réfléchissant, qui va recombinaison les deux faisceaux, et les envoyer vers deux détecteurs. La lumière agit comme une onde, et le montage est conçu de manière que les deux faisceaux se recombinent avec une différence de phase telle que les deux ondes s'annulent sur une des sorties, et se renforcent sur l'autre.

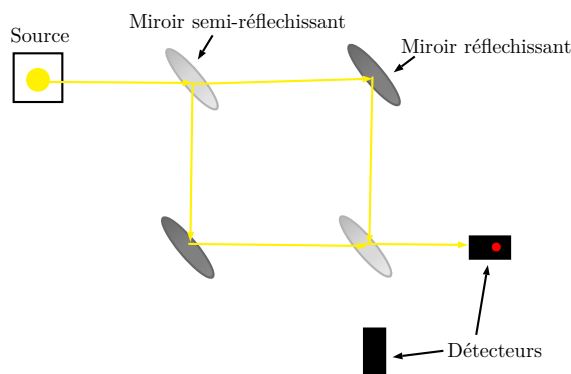


FIGURE 2.5 : Interféromètre de Mach-Zehnder

L'expérience devient intéressante lorsque l'on envoie les photons un par un, car on s'attendrait à ce qu'ils arrivent sur les deux détecteurs comme ils le feraient s'ils étaient des particules classiques, parce qu'ils ne devraient, selon notre intuition, parcourir qu'un seul des deux chemins et donc pouvoir atteindre les deux détecteurs, car ils ne devraient plus subir d'interférences. Or la réalité est différente, et on observe que les photons arrivent toujours sur le même détecteur, et donc que les photons interfèrent avec eux-mêmes, et qu'ils parcourent en quelque sorte les deux chemins en même temps.

Pour en revenir aux concepts évoqués précédemment, le premier miroir va créer une superposition quantique des deux états, qui sont les deux chemins possibles, et le deuxième miroir va impacter à nouveau cette superposition, et crée cette interférence quantique, qui mène à cette mesure étrange.

Le dernier concept que l'on va évoquer est celui de l'intrication quantique [27], qui se produit lorsque deux particules interagissent, et que l'on ne peut plus décrire l'état de chacune des particules indépendamment, mais que l'on doit décrire l'état du système formé par les deux particules. Si on reprend l'expérience de pensée du chat de Schrödinger, on peut imaginer que la vie du chat est liée à l'état de la particule radioactive, et donc que tant que l'on ne regarde pas dans la boîte, le chat est à la fois mort et vivant, et que l'on ne peut pas décrire l'état du chat sans décrire l'état de la particule radioactive. Cela peut paraître étrange, mais c'est un phénomène réel, et il a été mis en évidence par l'expérience d'Aspect, qui lui a valu le prix Nobel de physique en 2022.

Ce concept d'intrication fut mis en évidence par Schrödinger, puis Einstein, Podolsky et Rosen, ont émis l'hypothèse que la mécanique quantique était incomplète, et que les particules possédaient des propriétés cachées, car sinon cela impliquerait que les particules communiquent instantanément, ce qui est impossible selon la relativité restreinte, car cela impliquerait que l'information se déplace plus vite que la lumière. Néanmoins, cette simultanéité ne va pas à l'encontre de la relativité restreinte, via le théorème de non-communication, car pour constater cette causalité, il faudrait pouvoir communiquer de l'information, ce qui se fait à la vitesse de la lumière au maximum.

Néanmoins, l'hypothèse EPR (Einstein, Podolsky, Rosen) suggérait qu'il y avait des propriétés cachées, et donc que la mécanique quantique était incomplète, et donc qu'il existait une théorie plus fondamentale. C'est dans ce contexte que John Bell a énoncé son théorème, qui permet de tester expérimentalement l'hypothèse EPR, et qui consiste en une inégalité, qui si elle est

violée, implique qu'il y a une influence instantanée entre les particules et non un paramètre autre, et donc contredit l'hypothèse EPR. Alain Aspect a réalisé une expérience qui a violé cette inégalité de manière indiscutable [28], et qui de plus était en parfait accord avec la mécanique quantique.

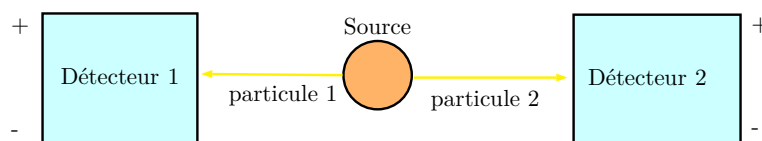


FIGURE 2.6 : Schéma d'un test EPR idéal

Cette expérience consiste en un laser, qui envoie des photons sur un cristal, qui va créer une paire de photons intriqués, qui vont être envoyés sur deux détecteurs, qui vont mesurer la polarisation des photons. Les deux détecteurs mesurent la polarisation des photons selon deux axes différents, et on peut choisir l'angle de ces axes, et donc on peut choisir la polarisation que l'on veut mesurer. On va changer cet angle après que les photons soient partis du cristal, et donc après qu'ils soient intriqués, et on va mesurer la polarisation des photons, et c'est cette polarisation qui va être corrélée, et qui va nous permettre de tester l'inégalité de Bell.

En résumé, la mécanique quantique est une théorie qui décrit le monde à l'échelle microscopique et qui fait apparaître des phénomènes qui n'ont pas d'équivalent en mécanique classique, et qui sont contre-intuitifs. Ceux-ci font intervenir des concepts probabilistes, de superposition quantique, d'intrication quantique, et d'interférence quantique, mais qui sont néanmoins prédictifs et vérifiables expérimentalement.

Cela fait de la physique quantique une théorie déterministe des systèmes considérés pour ce qui est de l'évolution temporelle, c'est-à-dire qui permet de décrire l'évolution d'un système à partir des conditions initiales, ce qui rend ses concepts applicables à l'informatique. Néanmoins, la mesure d'une observable va modifier l'état de façon aléatoire. On va donc perdre une partie de l'information lors de celle-ci, ce qui fait que les résultats sont probabilistes, et donc non purement déterministes à cet égard. De fait, on peut utiliser les propriétés de superposition des états quantiques afin de réaliser des calculs sur plusieurs états en même temps, et donc de manière parallèle, avec l'intrication qui permet de lier différents états entre eux et d'augmenter encore les possibilités de calculs. Tout cela permet de réaliser des opérations complexes, mais il faut garder en tête la perte d'information lors de la mesure, et les problèmes qui en découlent, car une interaction avec l'environnement peut détruire la fonction d'onde, tout comme une mesure, et donc détruire les calculs en cours.

Deuxième partie

Les notions de base

Chapitre 3

Un ordinateur classique

Avant de partir dans les concepts propres à l’algorithmie quantique, il est utile de présenter les concepts de base de l’algorithmie classique. Pour commencer, on va faire un tour des idées théoriques puis de l’implémentation matérielle.

3.1 Logique

Dans le cadre de l’informatique, on utilise la logique pour décrire le comportement d’un ordinateur. C’est une logique binaire, c’est-à-dire qu’on ne s’intéresse qu’à deux valeurs de vérité : vrai et faux. On la nomme logique booléenne, du nom de son inventeur, George Boole, et elle est très pratique dans le cadre de l’informatique, car elle permet de décrire simplement les opérations logiques de base.

En effet, les ordinateurs ont été conçus pour manipuler de telles valeurs, parce qu’elles peuvent être représentées par un concept simple de “on/off” (allumé/éteint), qui peut être représenté par un courant électrique. On utilisera plutôt les valeurs 1 et 0, mais le principe reste le même. On peut alors définir des opérations sur ces valeurs, qui sont les opérations logiques de base. Dans le cadre informatique, on les appelle les portes logiques [29], mais on peut également utiliser la notation mathématique de la logique afin de les décrire, ou encore les tables de vérité, qui décrivent le comportement de ces opérations sur toutes les valeurs possibles, ou même le langage naturel.

Définissons donc les opérations logiques de base, et leur comportement sur les valeurs de vérité. Tout d’abord, considérons une entrée A et une entrée B , qui peuvent prendre les valeurs 0 ou 1, ainsi qu’une sortie Q , qui prendra également les valeurs 0 ou 1. Tout d’abord prenons une seule entrée. Il est utile d’avoir une opération d’identité, qui ne change pas la valeur de l’entrée, que l’on nommera “Buffer”, qui pour une entrée A et une sortie Q se définit par $Q = A$. On peut également définir une opération de négation, qui inverse la valeur de l’entrée, que l’on nommera “NOT”, qui pour une entrée A et une sortie Q se définit par $Q = \neg A$.

Nom	Symbole	Table de vérité	
Buffer	$A \rightarrow Q$	A	Q
		0	0
		1	1
NOT	$A \rightarrow \neg Q$	A	Q
		0	1
		1	0

TABLE 3.1 : Résumé des opérations logiques sur une entrée

Mais on ne peut pas faire grand-chose avec une seule entrée. Ainsi, faisons de même avec deux, à commencer par la plus simple, qui est l'opération "AND", qui donne 1 si et seulement si les deux entrées sont à 1, et 0 sinon, et qui se définit par $Q = A \wedge B$. Le contraire de cette opération est l'opération "OR", qui donne 1 si au moins une des deux entrées est à 1, et 0 sinon, et qui se définit par $Q = A \vee B$. Pour ces deux portes, on peut également définir leur contraire, qui est l'opération "NAND" (NOT AND) et "NOR" (NOT OR), qui sont simplement les opérations AND et OR suivies d'une opération NOT. Finalement, on peut définir l'opération "XOR" (eXclusive OR), qui donne 1 si et seulement si une et une seule des deux entrées est à 1, et qui se définit par $Q = A \oplus B$, parfois également considérée comme la somme par bit. Similairement à NAND et NOR, on peut définir l'opération "XNOR" (eXclusive NOR), qui est simplement l'opération XOR suivie d'une opération NOT.

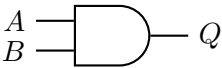





Nom	Symbole	Table de vérité															
AND		<table> <tr><th>A</th><th>B</th><th>Q</th></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	A	B	Q	0	0	0	0	1	0	1	0	0	1	1	1
A	B	Q															
0	0	0															
0	1	0															
1	0	0															
1	1	1															
OR		<table> <tr><th>A</th><th>B</th><th>Q</th></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	A	B	Q	0	0	0	0	1	1	1	0	1	1	1	1
A	B	Q															
0	0	0															
0	1	1															
1	0	1															
1	1	1															
XOR		<table> <tr><th>A</th><th>B</th><th>Q</th></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>0</td></tr> </table>	A	B	Q	0	0	0	0	1	1	1	0	1	1	1	0
A	B	Q															
0	0	0															
0	1	1															
1	0	1															
1	1	0															
NAND		<table> <tr><th>A</th><th>B</th><th>Q</th></tr> <tr><td>0</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>0</td></tr> </table>	A	B	Q	0	0	1	0	1	1	1	0	1	1	1	0
A	B	Q															
0	0	1															
0	1	1															
1	0	1															
1	1	0															
NOR		<table> <tr><th>A</th><th>B</th><th>Q</th></tr> <tr><td>0</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>0</td></tr> </table>	A	B	Q	0	0	1	0	1	0	1	0	0	1	1	0
A	B	Q															
0	0	1															
0	1	0															
1	0	0															
1	1	0															
XNOR		<table> <tr><th>A</th><th>B</th><th>Q</th></tr> <tr><td>0</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	A	B	Q	0	0	1	0	1	0	1	0	0	1	1	1
A	B	Q															
0	0	1															
0	1	0															
1	0	0															
1	1	1															

TABLE 3.2 : Résumé des opérations logiques sur deux entrées

À partir de ces opérations de bases, on peut construire tout ce que font les ordinateurs actuels. En effet, on peut définir des opérations plus complexes, comme l'addition, la soustraction, la multiplication, la division, etc. Mais cela demande un peu plus de travail, et on va juste montrer par exemple comment construire une addition de deux nombres binaires de 3 bits.

Afin de sommer deux nombres binaires, on peut les additionner bit à bit, en prenant en compte la retenue. Par exemple, pour additionner 5 (101) et 3 (011), cela donne :

$$\begin{array}{r}
 1 \quad 0 \quad 1 \\
 + \quad 0 \quad 1 \quad 1 \\
 \hline
 \end{array}
 \Rightarrow
 \begin{array}{r}
 \quad 0 \quad 1 \\
 + \quad 0 \quad 1 \quad 1 \\
 \hline
 \quad 0 \quad 0
 \end{array}
 \Rightarrow
 \begin{array}{r}
 1 \quad 1 \quad 1 \\
 + \quad 0 \quad 1 \quad 1 \\
 \hline
 0 \quad 0 \quad 0
 \end{array}
 \Rightarrow
 \begin{array}{r}
 1 \quad 1 \quad 1 \\
 + \quad 0 \quad 1 \quad 1 \\
 \hline
 1 \quad 0 \quad 0 \quad 0
 \end{array}$$

qui donne bien 8 (1000).

Alors, pour sommer simplement deux nombres binaires, on peut utiliser une porte XOR, qui donne 1 si et seulement si une et une seule des deux entrées est à 1, et une porte AND, qui donne 1 si et seulement si les deux entrées sont à 1. Cela reviendrait schématiquement à :

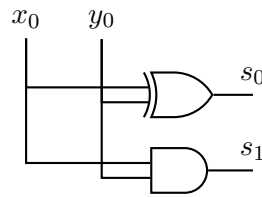


FIGURE 3.1 : Schéma d'une addition de deux bits

Mais afin ensuite de pouvoir additionner des nombres de plus de 1 bit, il faut prendre en compte la retenue. De fait on doit construire un circuit pour sommer trois bits, qui est le même sauf que l'on répète l'opération avec la première sortie et si l'une donne un reste, on le note, comme suit :

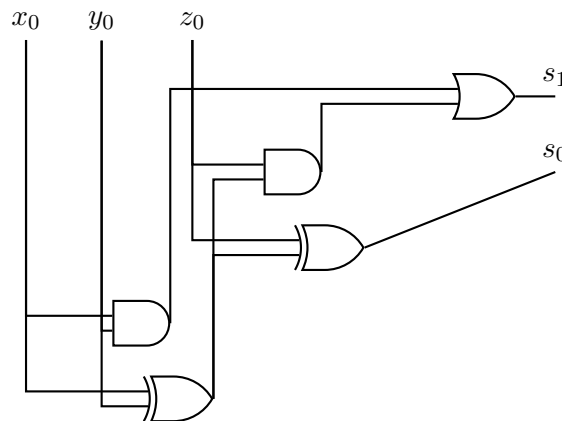


FIGURE 3.2 : Schéma d'une addition de trois bits

qui permet bien d'obtenir les quatre sorties de l'addition de trois bits, soit 00, 01, 10 et 11.

Une fois ces circuits de base construits, il est possible de les combiner pour obtenir un circuit permettant d'additionner deux nombres, par exemple de 3 bits chacun. Pour cela on procède exactement comme lorsque l'on a déconstruit l'addition plus haut, en sommant les bits de même poids, et en prenant en compte la retenue, via notre circuit de somme de trois bits.

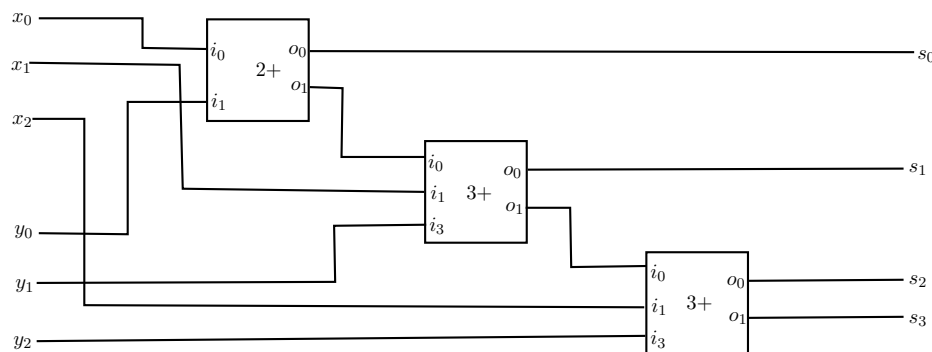


FIGURE 3.3 : Schéma d'une addition de deux nombres de 3 bits

On voit bien via cet exemple que même les choses les plus simples doivent être repensées pour être adaptées aux nouvelles technologies, que ce soit les ordinateurs classiques ou, comme on le verra plus tard, les ordinateurs quantiques.

Un dernier mot sur la notion d'universalité, qui se dit d'un ensemble de portes logiques qui permettent de construire n'importe quel circuit logique. En effet, cela revient à dire que matériellement, il suffit de pouvoir construire ces portes pour pouvoir effectuer les opérations que l'on souhaite. Dans le cadre des ordinateurs classiques, on peut montrer que les portes NAND et NOR sont universelles, mais on laissera la démonstration de ce fait pour les plus curieux, car elle n'est pas très compliquée à comprendre, mais assez longue.

3.2 Hardware

Comme vu juste avant, les ordinateurs classiques sont composés de portes logiques, qui peuvent toutes être construites à partir de portes NAND ou NOR. Afin de donner une idée du fonctionnement d'un ordinateur classique, on va détailler le fonctionnement pratique d'une implémentation de la porte NAND, qui est la plus simple à construire.

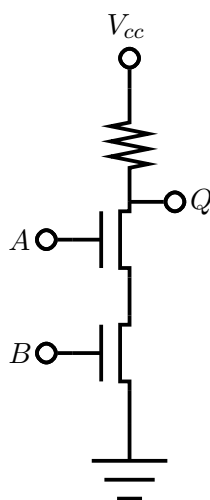


FIGURE 3.4 : Schéma électronique d'une porte NAND

On voit sur la figure 3.4 que la porte NAND est composée, de haut en bas, d'une résistance parcourue par un courant électrique de tension V_{cc} , qui mène d'un côté à une sortie Q et de l'autre à deux transistors alignés avec comme base A et B , finalement reliés à la mise à terre. De manière simplifiée, on peut dire que les transistors sont des interrupteurs qui laissent passer le courant lorsque la tension à leur base est suffisante. De plus, le fait de relier le circuit à la mise à terre met le circuit en état bas, soit le 0 logique. Ainsi, tant que les deux transistors ne sont pas activés, le courant ne peut pas passer, et la sortie est à 1, et dès que cela passe, ce dernier effet s'applique et la sortie passe à 0.

Mais cela n'est pas encore complètement satisfaisant, car il faut encore savoir construire le circuit. Néanmoins, seuls les transistors sont vraiment compliqués à construire, car les résistances sont relativement aisées à fabriquer étant une propriété naturelle des matériaux, et les mises à terre sont simplement des connexions à la terre, tout comme les connexions et la génération de tension qui sont les bases de l'électricité.

Pour décrire le fonctionnement d'un transistor, on va utiliser un modèle simplifié, qui est le modèle de transistor à effet de champ, ou FET (Field effect transistor) [30].

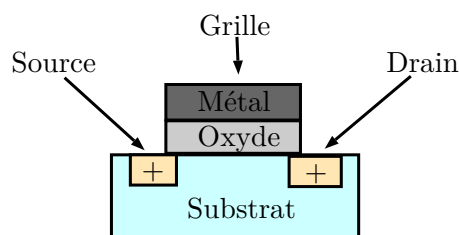


FIGURE 3.5 : Transistor à effet de champ (FET)

Le principe de fonctionnement est le suivant : lorsque la tension à la base de la grille est suffisante, le courant peut passer entre le drain et la source, et lorsque ce n'est pas le cas, le courant ne peut pas passer. La base, ou grille, est composée d'un métal qui est séparé du reste du transistor par une couche d'oxyde, qui est un isolant. Ainsi, lorsque la tension dans le métal est suffisante, cela va attirer ou repousser les électrons du substrat et permettre le passage du courant entre le drain et la source. Cela est possible car le substrat est un matériau semi-conducteur, qui est un matériau qui peut être isolant ou conducteur selon la tension qui lui est appliquée, généralement du silicium. Cette propriété découle de la structure de ce matériau, qui est composé d'atomes de silicium, qui ont 4 électrons de valence, et qui vont donc former une structure cristalline, où chaque atome est lié à 4 autres. Cependant, il manque un électron à chaque atome pour former une liaison covalente, et c'est ce qu'on appelle un trou. Ainsi, lorsqu'on applique une tension à la base, on va pouvoir déplacer les électrons et les trous, et ainsi modifier la conductivité du substrat. C'est une partie de la physique du solide nommée théorie des bandes [31], qui ne sera pas détaillée ici, mais mentionnée, car elle est aussi basée sur la mécanique quantique, ce qui explique l'utilisation de matériaux semi-conducteurs dans les ordinateurs quantiques.

Ainsi, on vient d'avoir un aperçu du fonctionnement d'un ordinateur classique, afin de pouvoir mieux comprendre les différences avec les ordinateurs quantiques, mais aussi les similarités entre les deux concepts. Un ordinateur classique est concrètement construit sur un circuit électronique dont les composants de base sont les transistors, qui sont construits à partir de matériaux semi-conducteurs, propriété qui est basée sur la mécanique quantique. Néanmoins, la logique utilisée en est indépendante, et est basée sur la logique booléenne, qui est une logique binaire, et donc purement déterministe.

Chapitre 4

Un qubit

L'informatique quantique est basée, similairement à l'informatique classique, sur un système pouvant prendre deux états, dénommés 0 et 1, que l'on nomme un qubit. La différence clé est que, dans le cas quantique, le système n'est pas décrit par la logique booléenne, mais suit les lois de la mécanique quantique. On a déjà vu un système à deux états, le spin dans la section 2.2. Comme dans ce cas, un qubit n'est pas dans un état bien défini par rapport à une observable donnée, mais peut être dans une superposition d'états, soit et dans l'état 0 et celui 1, chacun avec une probabilité bien définie.

4.1 Superposition d'états

Afin de décrire cette superposition du système, on utilise la notation de Dirac, ou notation bra-ket, qui est une notation mathématique permettant de décrire les états d'un système quantique. Nous n'introduisons ici que les principes utiles à la compréhension de l'informatique quantique. Comme présenté préalablement un système quantique peut être dans un état qui n'est pas défini par rapport à une observable donnée. Par exemple dans le cas d'un spin, si pour cet état on mesure le spin dans la direction z (définie par un système d'axes cartésiens) on observera soit le résultat $+1/2$ avec une probabilité $|\lambda_0|^2$ et, après la mesure, le spin sera dans la direction z notée 0, soit le résultat $-1/2$ avec une probabilité $|\lambda_1|^2$ et après la mesure le spin sera dans la direction $-z$ notée 1. On dira alors qu'avant la mesure le spin était dans un état de superposition de 0 et 1 avec un coefficient λ_0 et λ_1 . On va alors mettre ces coefficients dans un vecteur colonne, et nommé celui-ci un ket, noté $|\psi\rangle$:

$$|\psi\rangle = \begin{pmatrix} \lambda_0 \\ \lambda_1 \end{pmatrix}$$

Rappelons que les coefficients sont complexes, et que la somme des amplitudes doit être égale à 1, où l'amplitude est le carré du coefficient, soit $|\lambda_0|^2 + |\lambda_1|^2 = 1$.

Posons maintenant les états de base $|0\rangle$ et $|1\rangle$, qui sont les états dans lesquels le système peut être déterministiquement, soit $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ et $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. Ainsi, tout qubit d'état $|\psi\rangle$ peut être écrit comme une combinaison linéaire des états de base :

$$|\psi\rangle = \lambda_0 |0\rangle + \lambda_1 |1\rangle$$

On définit également le bra, noté $\langle\psi|$, comme le conjugué transposé du ket $|\psi\rangle$:

$$\langle\psi| = (\lambda_0^* \quad \lambda_1^*)$$

ce qui permet de définir le produit scalaire entre deux états $|u\rangle$ et $|v\rangle$ comme $\langle u|v\rangle$.

Via cette opération, on observe que l'on peut obtenir mathématiquement les coefficients d'un état en le multipliant par un état de base :

$$\begin{aligned}\langle 0|\psi\rangle &= \lambda_0 \langle 0|0\rangle + \lambda_1 \langle 0|1\rangle = \lambda_0 \\ \langle 1|\psi\rangle &= \lambda_0 \langle 1|0\rangle + \lambda_1 \langle 1|1\rangle = \lambda_1\end{aligned}$$

car $\langle 0|0\rangle = 1$ comme les états de base sont normalisés, et $\langle 0|1\rangle = 0$ car les états de base sont orthogonaux.

Notons que l'on pourrait choisir d'autres états de base, par exemple $|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ et $|-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$, qui sont orthogonaux, et donc forment une base ($\langle +|-\rangle = \left(\frac{1}{\sqrt{2}}\langle 0| + \frac{1}{\sqrt{2}}\langle 1|\right)\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{2}\langle 0|0\rangle - \frac{1}{2}\langle 0|1\rangle + \frac{1}{2}\langle 1|0\rangle - \frac{1}{2}\langle 1|1\rangle = 0$). Cette notion de base sera plus détaillée ultérieurement, ainsi que la dénomination $|+\rangle$ et $|-\rangle$.

4.2 Opérations sur un qubit

Similairement à l'informatique classique, on peut effectuer des opérations sur un qubit, que l'on nomme des portes [32]. Ces portes décrivent l'évolution du système, et similairement aux qubits, on peut les décrire via la notation de Dirac, mais également via de l'algèbre linéaire, comme des matrices unitaires, comme les coefficients d'un état sont multipliés par un nombre complexe de module 1, et donc la norme du vecteur reste la même. Cette section est aussi l'occasion de présenter les diagrammes de circuits quantiques, qui sont une représentation graphique des opérations effectuées sur un qubit.

La plus simple est comme toujours l'identité, qui ne fait rien, et est représentée par la matrice identité :

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

et son effet sur un qubit est de le laisser inchangé, soit $|\psi\rangle = I|\psi\rangle$, et se calcule via le produit matriciel :

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \lambda_0 \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} \lambda_0 \\ \lambda_1 \end{pmatrix}$$

qui revient schématiquement à juste avoir un qubit :

$$|\psi\rangle \text{ ———}$$

FIGURE 4.1 : Circuit quantique de l'identité

pour lequel on ne dessine aucune porte, car l'état ne change pas. Notons aussi que la porte peut être décrite par la notation de Dirac, soit $I = |0\rangle\langle 0| + |1\rangle\langle 1|$, car $I|\psi\rangle = (|0\rangle\langle 0| + |1\rangle\langle 1|)|\psi\rangle = |0\rangle\langle 0|\psi\rangle + |1\rangle\langle 1|\psi\rangle = |0\rangle\lambda_0 + |1\rangle\lambda_1 = |\psi\rangle$.

On va passer ensuite aux portes dites de Pauli [33], qui sont les portes de base de l'informatique quantique, et qui sont les matrices de Pauli. Celles-ci servent à la description du spin d'une particule, un système quantique à deux états, et il y en a trois, une pour chaque axe de l'espace, soit X , Y et Z .

La première est la porte X , qui est la plus simple car assimilable à la porte NOT de l'informatique classique, et est représentée par la matrice suivante :

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

ou encore $X = |0\rangle\langle 1| + |1\rangle\langle 0|$, et son effet sur un qubit est de permuter les coefficients de probabilité, soit $X|\psi\rangle = |1\rangle\langle 0|\psi\rangle + |0\rangle\langle 1|\psi\rangle = |1\rangle\lambda_0 + |0\rangle\lambda_1$ qui est bien l'inverse du qubit de départ. On représente cette porte par le circuit quantique suivant :

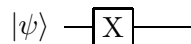


FIGURE 4.2 : Circuit de la porte X

La porte Z paraît un peu plus étrange, car elle change le signe, dit de phase, du coefficient de l'état 1, soit $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$, et son effet sur un qubit est $Z|\psi\rangle = (|0\rangle\langle 0| - |1\rangle\langle 1|)|\psi\rangle = |0\rangle\lambda_0 - |1\rangle\lambda_1$.

La troisième porte est la porte Y , qui semble structurellement à un mélange des deux précédentes, avec une unité imaginaire, soit $Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, et son effet sur un qubit est $Y|\psi\rangle = (i|0\rangle\langle 1| - i|1\rangle\langle 0|)|\psi\rangle = i|0\rangle\lambda_1 - i|1\rangle\lambda_0$.

Définissons la notion de vecteur propre afin d'expliquer une des notions de visualisation de l'informatique quantique. Un vecteur propre est un vecteur qui, lorsqu'il est multiplié par une matrice, donne un vecteur colinéaire au vecteur de départ. Posant une matrice M et un vecteur \vec{v} , on a $M\vec{v} = \lambda\vec{v}$, avec λ un scalaire.

Notons que les différentes portes de Pauli ont des vecteurs propres, pour X , on a $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ et $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, pour Y , on a $|\odot\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$ et $|\oslash\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)$, et pour Z , on a $|0\rangle$ et $|1\rangle$.

Démonstration. Montrons que $|+\rangle$ est un vecteur propre de X . On a $X|+\rangle = (|0\rangle\langle 1| + |1\rangle\langle 0|)\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle\langle 1|0\rangle + |0\rangle\langle 1|1\rangle + |1\rangle\langle 0|0\rangle + |1\rangle\langle 0|1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$, donc $|+\rangle$ est bien un vecteur propre de X . On peut faire de même pour les autres vecteurs propres. \square

Afin de visualiser l'état d'un qubit, on peut utiliser une sphère de Bloch [34], qui est une sphère unitaire, et qui est dans un espace à trois dimensions, avec les axes x , y et z qui correspondent aux vecteurs propres de X , Y et Z respectivement.

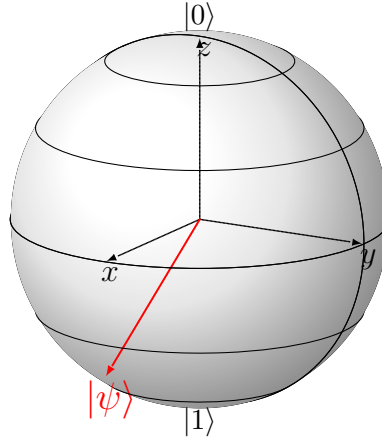


FIGURE 4.3 : Sphère de Bloch

L'effet des portes de Pauli sur la sphère de Bloch est de faire une symétrie par rapport à l'axe correspondant à la porte. Cette intuition par symétrie et rotation est très utile pour visualiser les portes quantiques.

La suivante est justement la porte de Hadamard, qui est définie par $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, et son effet sur la sphère de Bloch est de faire une rotation de 180° autour de l'axe y et de 90° autour de l'axe x . Cela à un effet intéressant sur les états de base, car $|0\rangle$ est envoyé sur $|+\rangle$ et $|1\rangle$ est envoyé sur $|-\rangle$.

Démonstration. On a $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ et $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. On a donc $H(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |0\rangle$, et un raisonnement similaire pour $|1\rangle$. \square

On remarque donc que cette porte permet de créer une superposition équilibrée. On note aussi que $H^2 = I$, donc H est sa propre inverse. On étudiera des effets plus concrets de cette porte dans la section 4.3.

Finalement, on évoquera rapidement les portes plus générales, les autres noms spécifiques de portes seront définis si besoin.

La première est la porte de phase, qui est définie par $P(\phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$. Elle a pour effet de changer la phase du qubit de ϕ , et est donc une généralisation de la porte Z en générant une rotation autour de l'axe z .

La seconde est la porte de rotation autour d'un axe sur la sphère de Bloch, qui est définie par

$$R_x = \begin{pmatrix} \cos \frac{\theta}{2} & -i \sin \frac{\theta}{2} \\ -i \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{pmatrix}$$

$$R_y = \begin{pmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{pmatrix}$$

$$R_z = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix}$$

qui ont pour effet de faire une rotation de θ autour de l'axe x , y et z respectivement (le travail de trigonométrie est laissé au lecteur, car l'exponentielle revient à cela par la formule d'Euler

$$e^{i\theta} = \cos \theta + i \sin \theta).$$

Notons qu'un qubit est par définition une combinaison linéaire $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, ce qui sans perte de généralité peut être écrit comme $|\psi\rangle = \cos \frac{\theta}{2} |0\rangle + e^{i\phi} \sin \frac{\theta}{2} |1\rangle$. Cela revient au même que de passer des coordonnées cartésiennes aux coordonnées sphériques, comme on peut le voir via la sphère de Bloch.

La porte générale sur un qubit est donc définie par des angles, qui ressemble à une combinaison de rotation autour des axes x , y et z :

$$U(\theta, \phi, \lambda) = \begin{pmatrix} \cos \frac{\theta}{2} & -e^{i\lambda} \sin \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} & e^{i(\phi+\lambda)} \cos \frac{\theta}{2} \end{pmatrix}$$

mais l'on se rend compte qu'il y a trois paramètres alors que plus haut, on a vu qu'il fallait deux angles pour définir un qubit. Néanmoins, on peut y donner une intuition, en appliquant la porte U à un qubit $|0\rangle$, on obtient bien $\cos \frac{\theta}{2} |0\rangle + e^{i\phi} \sin \frac{\theta}{2} |1\rangle$ comme énoncé plus haut, et sur $|1\rangle$, on obtient $e^{i\lambda}(-\sin \frac{\theta}{2} |1\rangle + e^{i\phi} \cos \frac{\theta}{2} |0\rangle)$. Le facteur devant $|1\rangle$ est $e^{i\lambda}$, qui est une phase, et donc n'a pas d'effet sur la sphère de Bloch, néanmoins n'étant présent que sur l'état $|1\rangle$, il va avoir un effet dès que l'on va traiter avec un état superposé.

Les portes se représentent par des carrés avec un symbole à l'intérieur, similairement à la porte X dans le cadre des diagrammes de circuits. De plus on peut noter que pour toutes les portes, il existe une porte inverse, qui est la porte qui annule l'effet de la porte, pour une porte générale U , sa porte inverse est U^\dagger et on a $UU^\dagger = I$.

4.3 Mesure

Une fois que le système a subi des transformations, il faut récupérer l'information. À la différence des opérations précédentes, la mesure est une opération irréversible, et qui va détruire l'information contenue dans le qubit en vertu de la mécanique quantique. Elle peut être interprétée comme la projection du vecteur état initial sur le vecteur de la sphère de Bloch correspondant à la direction de la mesure ce qui fait perdre une partie de l'information.

De plus, lorsque l'on mesure un qubit en pratique, on va répéter l'expérience plusieurs fois, et on va selon les cas obtenir des résultats différents qu'on va pouvoir analyser statistiquement.

Prenons l'exemple d'un qubit dans l'état $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, obtenu par l'application de la porte H sur le qubit $|0\rangle$. La théorie de la physique quantique décrit que si on mesure ce qubit, on a une chance sur deux d'obtenir $|0\rangle$ et une chance sur deux d'obtenir $|1\rangle$ (car les deux ont un coefficient de $\frac{1}{\sqrt{2}}$ qui donne une probabilité de $(\frac{1}{\sqrt{2}})^2 = \frac{1}{2}$).

On enregistre le résultat de la mesure dans un bit classique, et tout le système peut être représenté par le schéma suivant :

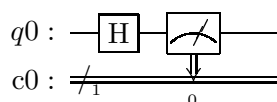


FIGURE 4.4 : Circuit de mesure d'un qubit dans l'état $|+\rangle$

Si on le fait sur une vraie machine quantique, on obtiendra parfois 0, parfois 1, dans la mesure effectuée dans l'exemple, 0 a été obtenu 3901 fois et 1 a été obtenu 4291 fois, ce qui nous donne

une probabilité proche de ce qu'on attendait. On peut émettre plusieurs hypothèses sur les raisons de cette différence, comme le fait que la machine quantique soit sensible à l'environnement et donc pas parfaite, ou que l'échantillon soit trop petit pour avoir une statistique parfaite. Cela peut être utile de représenter les résultats obtenus sous forme d'histogramme, surtout lorsque l'on va effectuer des mesures plus complexes.

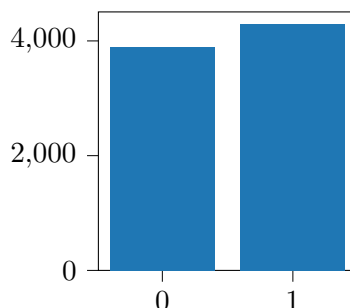


FIGURE 4.5 : Histogramme des résultats de la mesure d'un qubit dans l'état $|+\rangle$ ¹

De plus, la base utilisée par convention est la base $|0\rangle$ et $|1\rangle$, soit la base nommée z . Néanmoins, cela peut être intéressant de mesurer dans une autre base, par exemple la base x qui est la base $|+\rangle$ et $|-\rangle$. Parfois cette mesure dans une autre base est faite avec l'ordinateur, mais on peut aussi la faire avec des portes quantiques, par exemple pour mesurer dans la base x , on applique la porte H avant de mesurer.

Un qubit est donc un système quantique à deux états. De cela, on peut exploiter ses propriétés pour faire des opérations, telles que la superposition d'états. Ces opérations sont purement déterministes, et on peut les représenter par des portes quantiques. Cet état déterministe va être perdu lors de la mesure, qui va donner un résultat aléatoire selon l'état du qubit, et qui va détruire l'information contenue dans le qubit.

¹Exécuté le 29.08.2023 sur la machine 'ibm_quito', job id : cjmo77kvcjlr5ddj7h0

Chapitre 5

Système à plusieurs qubits

Une fois que l'on a compris le fonctionnement d'un qubit, il est pratique, sinon indispensable, de pouvoir travailler avec plusieurs qubits. En effet, pour le moment, nous ne pouvons rien faire de bien utile outre créer des résultats pseudo-aléatoires. La manipulation de plusieurs qubits permet de créer des états d'intrication, qui sont la base de la puissance des ordinateurs quantiques, et qui permettent en quelque sorte de faire des calculs en parallèle.

5.1 Description du système

Un système à plusieurs qubits est décrit similairement à un système à un seul qubit. Considérons deux qubits, $|\psi\rangle = \psi_1 |0\rangle + \psi_2 |1\rangle$ et $|\phi\rangle = \phi_1 |0\rangle + \phi_2 |1\rangle$. Le système est décrit par le produit tensoriel des deux qubits, soit $|\psi\rangle \otimes |\phi\rangle = \psi_1 \phi_1 |00\rangle + \psi_1 \phi_2 |01\rangle + \psi_2 \phi_1 |10\rangle + \psi_2 \phi_2 |11\rangle = |\psi\phi\rangle$. Cela revient donc à considérer un seul qubit à 4 états, $|00\rangle$, $|01\rangle$, $|10\rangle$ et $|11\rangle$, et les coefficients sont obtenus en multipliant les coefficients des deux qubits. Les probabilités sont obtenues en prenant le module au carré des coefficients, comme précédemment.

La logique est la même pour un système à n qubits, que l'on considère comme un seul vecteur à 2^n états. Cette augmentation du nombre d'états est la raison pour laquelle les ordinateurs quantiques sont théoriquement intéressants, et de plus la complexité des simulations classiques, car elle augmente exponentiellement avec le nombre de qubits.

Le formalisme de Dirac permet de mieux comprendre la notion d'intrication. On parle d'intrication lorsque l'on ne peut pas décrire un système par le produit de qubits individuels. Par exemple, considérons le système $|\psi\rangle = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle$. En reprenant l'égalité vue précédemment, $|\psi\rangle \otimes |\phi\rangle = \psi_1 \phi_1 |00\rangle + \psi_1 \phi_2 |01\rangle + \psi_2 \phi_1 |10\rangle + \psi_2 \phi_2 |11\rangle$, essayons de trouver les coefficients ψ_1 , ψ_2 , ϕ_1 et ϕ_2 . En comparant les facteurs, on obtient un système d'équations :

$$\begin{cases} |00\rangle : \psi_1 \phi_1 &= \frac{1}{\sqrt{2}} \\ |01\rangle : \psi_1 \phi_2 &= 0 \\ |10\rangle : \psi_2 \phi_1 &= 0 \\ |11\rangle : \psi_2 \phi_2 &= \frac{1}{\sqrt{2}} \end{cases}$$

On voit que ψ_1 ou ϕ_1 devrait être nul pour qu'il soit possible de définir les états respectifs des deux sous-systèmes, or si l'on prend les autres équations, la première implique que ψ_1 est non nul, et la dernière que ϕ_1 est non nul. On observe donc que le système n'est pas descriptible par le produit de deux qubits individuels, et on dit qu'il est intriqué.

Le produit tensoriel s'applique aussi aux matrices, et on peut donc définir similairement l'application d'une porte sur un système à plusieurs qubits. Par exemple si l'on souhaite appliquer une porte d'Hadamard sur le premier qubit d'un système à deux qubits, on obtient une porte globale sur le système :

$$H \otimes I = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}$$

que l'on peut appliquer sur le système afin de trouver le nouveau vecteur d'état.

5.2 Les portes

Mais l'intérêt des systèmes à plusieurs qubits n'est pas de faire ce que l'on sait déjà faire avec un seul qubit, mais de pouvoir faire des choses nouvelles.

La première porte que l'on va voir est la porte de CX (Controlled X, parfois aussi CNOT pour Controlled NOT), qui est une porte à deux qubits. Elle va prendre en entrée deux qubits, un qubit de contrôle et un qubit cible. Si le qubit de contrôle est à 0, la porte ne fait rien, sinon elle applique une porte X sur le qubit cible. Elle est représentée par la matrice suivante :

$$CX = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

et on peut la représenter sur un circuit comme suit :

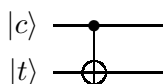


FIGURE 5.1 : Circuit d'une porte CX

avec $|c\rangle$ le qubit de contrôle et $|t\rangle$ le qubit cible. C'est une porte très importante, car elle permet de créer de l'intrication, de plus elle agit comme une porte XOR entre les deux qubits, avec le deuxième qubit comme sortie, soit symboliquement $|c, t\rangle \rightarrow |c, c \oplus t\rangle$ (le qubit cible va être modifié comme s'il subissait une porte XOR entre lui et le qubit de contrôle).

L'autre porte sur deux qubits que l'on va le plus utiliser est la porte CZ (Controlled Z). Elle est similaire à la porte CX, mais au lieu d'appliquer une porte X sur le qubit cible, elle applique une porte Z. Ce qui est particulier avec cette porte, c'est qu'elle agit sur tout le système, et pas seulement sur le qubit cible. En effet, si le qubit de contrôle est à $|0\rangle$, la porte ne fait rien, sinon elle applique une porte Z sur le qubit cible, ce qui revient à multiplier le vecteur d'état par -1 . Or, si le qubit cible est à $|0\rangle$, on a $|0\rangle \rightarrow -|0\rangle = |0\rangle$ et donc l'état du système ne change pas. En résumé, cette porte inverse la phase globale du système si et seulement si les deux qubits sont à $|1\rangle$. C'est pour cela qu'on la dessine comme suit :

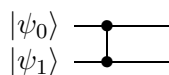


FIGURE 5.2 : Circuit d'une porte CZ

et cela est également visible sur sa matrice :

$$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Finalement, un paterne apparait et on peut noter la représentation générale d'une porte contrôlée U comme suit :

$$CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{11} & u_{12} \\ 0 & 0 & u_{21} & u_{22} \end{pmatrix}$$

où $U = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}$ est la matrice de la porte U .

Cette porte générale donne aussi une idée de comment sont construites les matrices des portes sur plusieurs qubits. En effet, on remarque que les lignes correspondent à l'effet de la porte sur les états de base $|00\rangle$, $|01\rangle$, $|10\rangle$ et $|11\rangle$, et les colonnes, ce en quoi la porte transforme les états de base. Par exemple, sur la dernière matrice présentée, on voit que les deux premières lignes ont juste des 1 dans la colonne correspondante, ce qui indique que pour les états $|00\rangle$ et $|01\rangle$, la porte ne fait rien. Sur les deux dernières lignes, on a des éléments uniquement sur les deux dernières colonnes, ce qui indique que le premier qubit demeure à 1, et que le deuxième subit une porte U .

Notons encore une porte sur trois qubits, la porte CCX (Controlled Controlled X), ou porte de Toffoli. Le principe est le même que pour la porte CX, mais avec deux qubits de contrôle, et la porte X est appliquée sur le qubit cible si et seulement si les deux qubits de contrôle sont à 1.

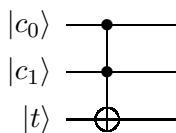


FIGURE 5.3 : Circuit d'une porte de Toffoli

De nombreuses autres portes existent, mais elles ne valent pas la peine d'être mentionnées ici, car elles ne seront pas utilisées dans ce travail, ou de manière assez succincte pour être expliquées au fur et à mesure.

5.3 Applications simples

Ci-après deux exemples de circuits visant à illustrer certains avantages de l'informatique quantique. D'un côté, le protocole de superdense coding permet d'envoyer deux bits d'information en n'envoyant qu'un seul qubit, et de l'autre, le protocole de quantum teleportation qui transfère l'état d'un qubit sans transférer le qubit lui-même, en utilisant deux bits classiques. Ceux-ci permettent de saisir l'avantage intrinsèque de l'informatique quantique, dans ces exemples où un qubit correspond à deux bits classiques, mais aussi de voir comment l'intrication peut être utilisée pour transmettre de l'information.

Mais avant de commencer, il faut savoir comment créer un état intriqué, comme décrit dans la section 5.1. On va expliquer la création d'un état intriqué dit équilibré, soit avec la même probabilité d'être dans les différents états intriqués, pour un système de deux qubits un tel état est nommé un état de Bell. Restons sur le système de deux qubits, sur lequel on compte quatre états de Bell :

$$\begin{aligned} |\Phi^+\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\ |\Phi^-\rangle &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \\ |\Psi^+\rangle &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \\ |\Psi^-\rangle &= \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \end{aligned}$$

On va montrer comment créer l'état $|\Phi^+\rangle$, et les autres états sont créés de manière similaire. Le circuit est le suivant :

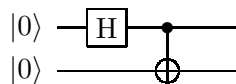


FIGURE 5.4 : Circuit de création de l'état $|\Phi^+\rangle$

qui dont on se rend compte assez intuitivement qu'il a l'effet escompté. En effet, on applique une porte de Hadamard sur le premier qubit, ce qui le met dans un état superposé, et ensuite, on applique une porte CX , qui va mettre le deuxième qubit dans le même état que le premier, et donc les deux qubits sont dans le même état, et sont intriqués. La porte de Hadamard fait changer le système $|00\rangle \rightarrow \frac{|0\rangle+|1\rangle}{\sqrt{2}} |0\rangle$ puis la porte CX applique comme une porte XOR , donc on obtient $\frac{|0\rangle+|1\rangle}{\sqrt{2}} |0\rangle \rightarrow \frac{|0,0\rangle+|1,1\rangle}{\sqrt{2}} = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$.

On peut maintenant passer aux deux exemples en gardant en tête ce qui a été dit précédemment.

5.3.1 Superdense Coding

Le principe du superdense coding [35] est d'envoyer deux bits d'information en n'envoyant qu'un seul qubit. On va donc considérer trois personnes, Alice qui veut envoyer un message à Bob, et Charlie qui va créer l'état intriqué. Une fois son travail terminé, Charlie envoie un qubit à

Alice, et un à Bob. Alice va pouvoir ensuite encoder son message dans le qubit qu'elle a reçu, selon ce que l'on va voir, et ensuite, elle va envoyer son qubit à Bob. Bob va ensuite pouvoir décoder le message en appliquant les portes adéquates sur la paire de qubits qu'il a reçu.

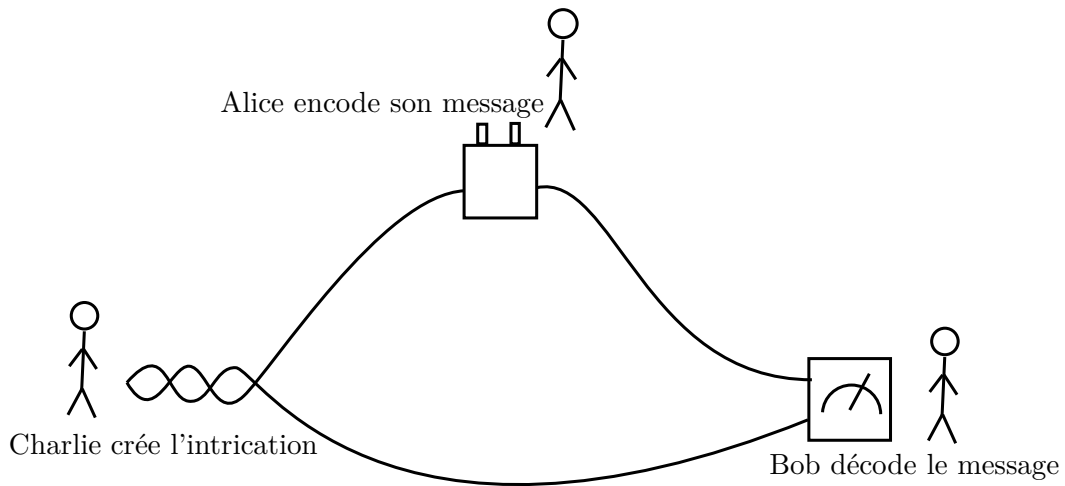


FIGURE 5.5 : Schéma du principe de superdense coding

Le circuit de Charlie est celui de la figure 5.4, afin de créer un état intriqué $|\phi^+\rangle$ duquel il envoie un qubit à Alice, et un à Bob.

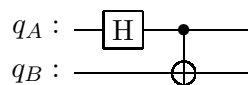


FIGURE 5.6 : Circuit de Charlie

De l'autre côté, celui de Bob va être l'inverse de celui de Charlie, afin de décoder le message. Cela est possible, car comme dit plus tôt, les portes sont réversibles. Ensuite, il va pouvoir mesurer afin de récupérer les deux bits d'information.

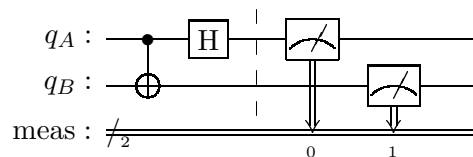


FIGURE 5.7 : Circuit de Bob

Il reste à montrer comment Alice encode son message. Alice reçoit le qubit q_A . Sur le circuit tel qu'on la construit, Alice va influencer du point de vue mathématique le qubit à droite (donc initialement on a $|00\rangle = |0\rangle_B \otimes |0\rangle_A$). Après le passage dans la partie de Charlie, le circuit est dans l'état $|\Phi^+\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$. Pour encoder son message, Alice a alors plusieurs possibilités :

- Si elle veut envoyer le message 00, elle ne fait rien, et le circuit reste dans l'état $|\Phi^+\rangle$.

- Similairement à l'ordinateur classique, elle peut inverser son qubit via une porte X , et le circuit passe dans l'état $\frac{|01\rangle+|10\rangle}{\sqrt{2}}$. Avec la mesure de Bob, on obtient alors 10.
- L'avantage quantique est dans les deux derniers, où Alice peut encoder de l'information dans la phase du qubit. En effet, via la porte Z elle peut l'inverser. Ainsi le circuit passe dans l'état $\frac{|00\rangle-|11\rangle}{\sqrt{2}}$ qui va donner 01 à Bob.
- Enfin, elle peut inverser le qubit et inverser la phase, ce qui donne $\frac{|01\rangle-|10\rangle}{\sqrt{2}}$ et donc 11 à Bob.

On peut vérifier les affirmations sur ce que recevra Bob en faisant le calcul. Par exemple,

$$\frac{|01\rangle - |10\rangle}{\sqrt{2}} \rightarrow \frac{|01\rangle - |11\rangle}{\sqrt{2}} \text{ après la } CX, \text{ et finalement } \frac{|01\rangle - |11\rangle}{\sqrt{2}} \rightarrow \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ 0 \\ -1 \end{pmatrix} =$$

$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = |11\rangle$ après la H . Il en va de même pour les autres cas.

On peut résumer toutes les étapes en un circuit.

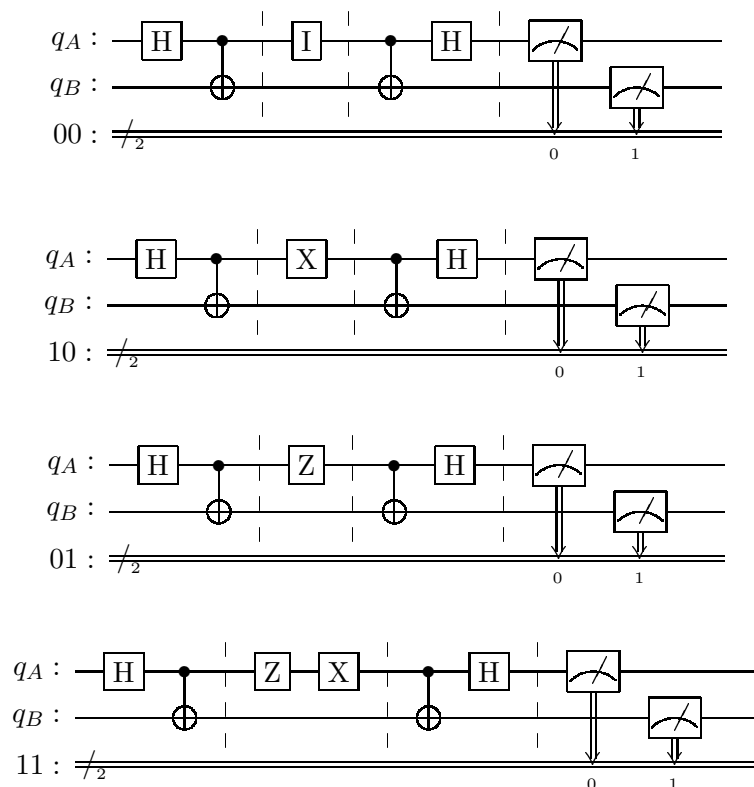


FIGURE 5.8 : Circuit complet : Charlie | Alice | Bob (circuit | mesure)

Testons maintenant l'un de ces circuits avec un vrai ordinateur quantique.

Comme vu avec le calcul, on est censé obtenir exactement la bonne valeur à chaque fois. Pourtant, les résultats obtenus ne semblent pas tout à fait d'accord avec la théorie. Ci-suit l'histogramme des résultats obtenus avec le circuit qui code le message 01 (voir figure 5.8).

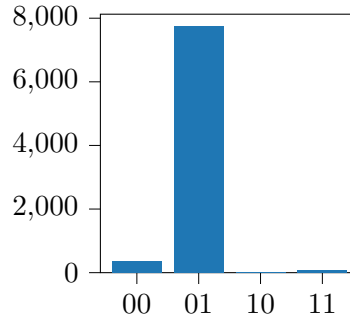


FIGURE 5.9 : Histogramme des résultats obtenus avec le circuit qui code le message 01 ¹

On se rend compte qu'il y a des résultats incorrects. Parmi les 8192 essais, 7739 sont corrects, ce qui fait un taux de réussite de 94.5%. Les autres forment le bruit, qui sont les erreurs qui arrivent lors du processus. Cela illustre très bien le problème des ordinateurs quantiques actuels : ils sont très sensibles aux erreurs. Nous parlerons par la suite des causes et voie de correction de ces erreurs.

5.3.2 Téléportation quantique

Il existe en physique quantique un théorème de non-clonage [36] qui dit qu'il est impossible de copier à l'identique un état quantique inconnu. Notons que deux états intriqués ne peuvent pas être considérés comme des copies de part leur mesure qui est identique.

Démonstration. Soient A et B deux systèmes quantiques, avec A dans un état $|a\rangle_A$ que nous souhaitons cloner et B dans un état $|b\rangle_B$ quelconque. Le système total est donc dans l'état $|a\rangle_A \otimes |b\rangle_B = |a\rangle_A |b\rangle_B$.

On ne peut pas copier l'état $|a\rangle_A$ en le mesurant, car cela détruirait une partie de l'information. On ne peut donc qu'agir sur son opérateur d'évolution U (les portes quantiques sont de tels opérateurs). On cherche U tel que $U |a\rangle_A |b\rangle_B = |a\rangle_A |a\rangle_B$. Cela implique également que pour un état quelconque $|\alpha\rangle_A$ de A , $U |\alpha\rangle_A |b\rangle_B = |\alpha\rangle_A |\alpha\rangle_B$.

Rappelons que les opérateurs d'évolution sont unitaires, donc $U^\dagger U = I$, et que $|\psi\rangle^\dagger = \langle\psi|$, ainsi que $M^\dagger N^\dagger = (NM)^\dagger$. Étudions donc le produit scalaire de $|a\rangle_A |b\rangle_B$ avec $|\alpha\rangle_A |b\rangle_B$:

$$\begin{aligned} \langle b|_B \langle a|_A |\alpha\rangle_A |b\rangle_B &= \langle b|_B \langle a|_A U^\dagger U |\alpha\rangle_A |b\rangle_B = (U |a\rangle_A |b\rangle_B)^\dagger (U |\alpha\rangle_A |b\rangle_B) \\ &= (|a\rangle_A |a\rangle_B)^\dagger (|\alpha\rangle_A |\alpha\rangle_B) = \langle a|_B \langle a|_A |\alpha\rangle_A |\alpha\rangle_B \end{aligned}$$

Donc $\langle b|_B \langle a|_A |\alpha\rangle_A |b\rangle_B = \langle a|_B \langle a|_A |\alpha\rangle_A |\alpha\rangle_B \Leftrightarrow \langle a|\alpha\rangle = \langle a|\alpha\rangle^2$, ce qui est possible que si $\langle a|\alpha\rangle = 0$ ou 1, donc que ces états sont identiques ou orthogonaux.

On a donc une contradiction avec l'hypothèse de départ supposant ces états quelconques. On a donc montré qu'il est impossible de cloner un état quantique inconnu par l'absurde. \square

Ce théorème pose différents problèmes, notamment pour créer des méthodes de suppression d'erreurs, car on ne peut pas copier le système pour le comparer avec l'original.

¹Exécuté le 29.08.2023 sur la machine 'ibm_quito', job id : cjmvmniijvusg3savs30

Mais si la copie est impossible, on peut transmettre un état quantique en utilisant un état intriqué et des bits classiques [37].

Prenons à nouveau nos trois personnages : Alice, Bob et Charlie. Charlie va toujours créer un état intriqué, et envoyer un qubit à Alice et un à Bob. Alice va ensuite appliquer des opérations sur son qubit et envoyer le résultat à Bob, via un canal classique de communication, puis Bob va appliquer des opérations sur son qubit afin de retrouver ce qu'Alice lui a envoyé.

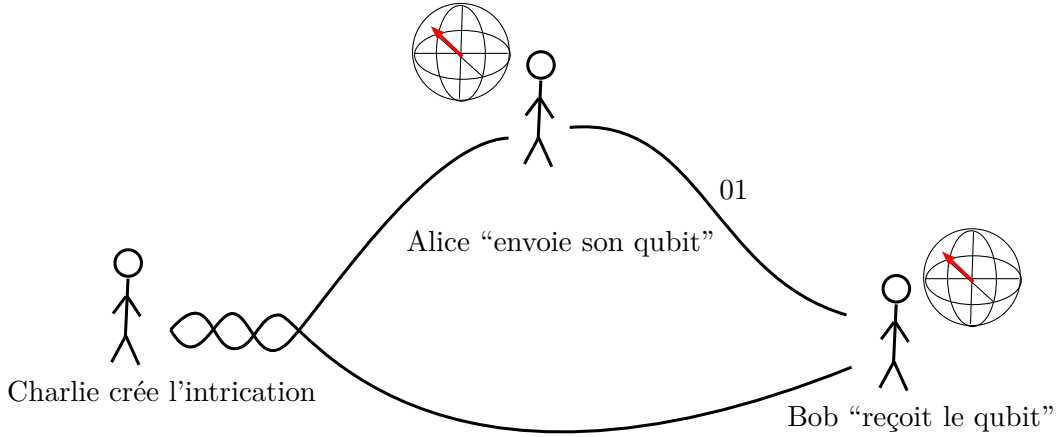


FIGURE 5.10 : Schéma du protocole de téléportation quantique

Voyons donc concrètement comment cela fonctionne. Le circuit à décortiquer est le suivant :

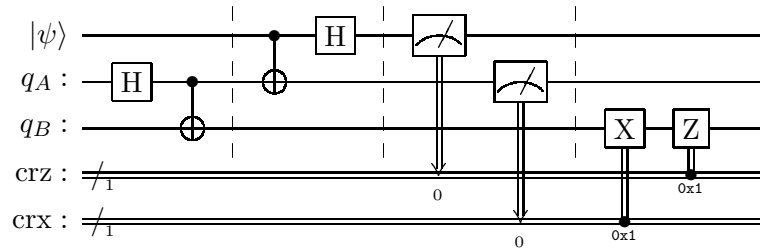


FIGURE 5.11 : Circuit complet : Charlie | Alice (opération | mesure) | Bob

Alice veut transmettre l'état $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ à Bob. Charlie va donc créer un état intriqué $|e\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, puis transmettre le premier qubit q_A à Alice et le second q_B à Bob. Gardons en tête donc que le qubit le plus à droite est celui de Bob, et celui le plus à gauche celui d'Alice $|e\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B)$.

Avec celui $|\psi\rangle$ que l'on veut transmettre, on a donc un système de trois qubits : $|\psi\rangle \otimes |e\rangle = \frac{1}{\sqrt{2}}(\alpha|0\rangle \otimes (|00\rangle + |11\rangle) + \beta|1\rangle \otimes (|00\rangle + |11\rangle)) = \frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|011\rangle + \beta|100\rangle + \beta|111\rangle)$.

En appliquant le porte CX avec le qubit de gauche comme contrôle et celui du milieu comme cible, on obtient : $\frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|011\rangle + \beta|100\rangle + \beta|111\rangle) \rightarrow \frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|011\rangle + \beta|110\rangle + \beta|101\rangle)$.

En appliquant la porte H sur le qubit de gauche, le facteur devant devient $\frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} = \frac{1}{2}$, et les termes $|0ij\rangle \rightarrow |0ij\rangle + |1ij\rangle$, et $|1ij\rangle \rightarrow |0ij\rangle - |1ij\rangle$, donc on obtient : $\frac{1}{2}(\alpha(|000\rangle + |011\rangle + |100\rangle + |111\rangle) + \beta(|010\rangle + |010\rangle - |110\rangle - |101\rangle))$.

Ceci peut être réécrit de la manière suivante :

$$\begin{aligned} \frac{1}{2} (& |00\rangle (\alpha |0\rangle + \beta |1\rangle) \\ & + |01\rangle (\alpha |1\rangle + \beta |0\rangle) \\ & + |10\rangle (\alpha |0\rangle - \beta |1\rangle) \\ & + |11\rangle (\alpha |1\rangle - \beta |0\rangle)) \end{aligned}$$

Cela permet de voir que la mesure d'Alice va projeter le qubit de Bob dans l'un des quatre états suivants :

- $|00\rangle \rightarrow \alpha |0\rangle + \beta |1\rangle$
- $|01\rangle \rightarrow \alpha |1\rangle + \beta |0\rangle$
- $|10\rangle \rightarrow \alpha |0\rangle - \beta |1\rangle$
- $|11\rangle \rightarrow \alpha |1\rangle - \beta |0\rangle$

et donc Bob n'a plus qu'à appliquer les portes X et Z selon les bits reçus pour retrouver l'état $|\psi\rangle$.

Alice mesure le qubit de droite sur le registre crx , et donc la porte X est appliquée uniquement si le bit est à 1, ce qui va inverser les 0 et les 1. De même pour le signe, mais avec celui de gauche et le registre crz , ainsi qu'avec la porte Z .

On peut également le faire sans mesure si l'on souhaite mettre un autre qubit dans l'état désiré en appliquant des portes CX et CZ depuis les bits que l'on mesurait précédemment.

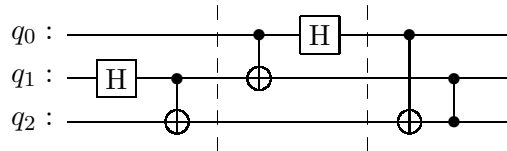


FIGURE 5.12 : Téléportation quantique sans mesure

Ce circuit a un intérêt de démonstration de la valeur d'un qubit question donnée, mais n'est pas très intéressant à utiliser en tant que tel, c'est pour cela qu'on le désigne comme un protocole.

En résumé, on a vu deux circuits mettant en évidence des avantages et complications liés à la nature quantique des qubits. En plus de donner une idée de l'importance des concepts de superposition et d'intrication, ces circuits permettent de voir que les qubits peuvent transporter plus d'un bit d'information par qubit en quelque sorte.

Chapitre 6

Un ordinateur quantique

Après avoir présenté la théorie, on va montrer comment cela a été implémenté dans la réalité. Notons que ce qui va suivre est principalement pour donner une idée, et non une description complète et précise de la réalité et des techniques utilisées.

6.1 Universalité

La notion d'universalité a déjà été évoquée plus haut, et donc tout comme en informatique classique, on va chercher à trouver un ensemble de portes permettant de réaliser n'importe quelle opération, dans ce cas avec une précision arbitraire. Cette limitation de précision est une des raisons de l'apparition d'erreur et de la nécessité de correction d'erreur.

Il existe plusieurs ensembles de portes universels [38], mais on va se concentrer sur l'ensemble de portes universel de $T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$, H et CX .

Tout d'abord, on va montrer que T et H sont universels permettant de réaliser n'importe quelle opération sur un seul qubit [39, 40]. Premièrement, on va montrer que T et H [41] permettent de réaliser une rotation d'angle arbitraire autour d'un axe \hat{n} .

Pour commencer, on a $HTH = R_x(\pi/4)$ et rappelons que $T = R_z(\pi/4)$, ce qui permet de construire la porte $R_{\hat{n}} = R_z(\pi/4)R_x(\pi/4) = e^{-i\frac{\pi}{8}Z}e^{-i\frac{\pi}{8}X} = \cos^2\frac{\pi}{8}I - i(\cos\frac{\pi}{8}(X+Z) + \sin\frac{\pi}{8}Y)\sin\frac{\pi}{8}$ (notons que $Y = -iZX$).

C'est une rotation d'axe $\hat{n} = \begin{pmatrix} \cos\frac{\pi}{8} \\ \sin\frac{\pi}{8} \\ \cos\frac{\pi}{8} \end{pmatrix}$ sur la sphère de Bloch, avec un angle de rotation donné

par $\cos\frac{\theta}{2} = \cos^2\frac{\pi}{8} \Rightarrow \theta = 2\arccos(\cos^2\frac{\pi}{8}) \approx 1.096$ qui n'est pas un multiple rationnel de 2π . On peut donc réaliser une rotation d'angle arbitraire autour de l'axe \hat{n} en appliquant $R_{\hat{n}}$ un nombre arbitraire de fois, car on n'atteindra jamais un angle multiple de 2π . Posons alors $R_{\hat{n}}(\alpha) = R_{\hat{n}}^{n_1}$ avec $n_1 \in \mathbb{N}$.

On peut également construire $HR_{\hat{n}}(\alpha)H = R_{\hat{m}}(\alpha)$ avec $\hat{m} = \begin{pmatrix} \cos\frac{\pi}{8} \\ -\sin\frac{\pi}{8} \\ \cos\frac{\pi}{8} \end{pmatrix}$ donc on peut réaliser

une rotation d'angle arbitraire autour de l'axe \hat{m} .

Finalement, on peut prouver que n'importe quelle porte U peut être écrite comme

$$U = R_{\hat{n}}(\theta_1)R_{\hat{m}}(\theta_2)R_{\hat{n}}(\theta_3)$$

et donc n'importe quelle porte sur un seul qubit peut être réalisée avec une précision arbitraire avec T et H .

La porte CX [42, 43] permet de réaliser une porte CU quelconque, car on peut créer déjà n'importe quelle porte sur un seul qubit [44].

Prenons une manière d'écrire une porte générale sur un seul qubit $U(\alpha, \beta, \gamma, \delta)$ comme suit

$$U = e^{i\alpha} \begin{pmatrix} e^{-i\frac{\beta+\delta}{2} \cos \frac{\gamma}{2}} & -e^{-i\frac{\beta-\delta}{2} \sin \frac{\gamma}{2}} \\ e^{i\frac{\beta-\delta}{2} \sin \frac{\gamma}{2}} & e^{i\frac{\beta+\delta}{2} \cos \frac{\gamma}{2}} \end{pmatrix}$$

qui permet de réécrire U comme $U = e^{i\alpha} AXBXC$ avec X la porte de Pauli X et A, B, C des portes sur un seul qubit, dont on peut passer le détail de leur construction algébrique, mais qui peuvent être précisément déterminées et donc implémentées.

En posant aussi $D = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$, on peut construire la porte CU comme suit

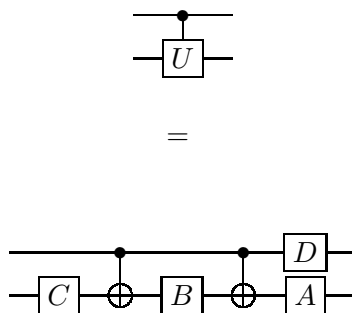


FIGURE 6.1 : Circuit d'une porte CU

où l'on voit la porte D sur le qubit de contrôle, car la phase globale de la porte U va impacter le qubit de contrôle, et doit donc être prise en compte par l'ajout de la porte D .

On peut ensuite se convaincre qu'à partir de n'importe quelle porte U sur un seul qubit, et porte contrôlée CU , on peut construire n'importe quelle porte sur n qubits. La démonstration est néanmoins trop complexe pour être présentée ici.

6.2 Hardware

La création d'ordinateurs quantiques est un défi technologique majeur, car il faut pouvoir manipuler des qubits, qui sont des systèmes quantiques, avec une précision extrême, et ce, à grande échelle.

Plusieurs facteurs compliquent la tâche, comme la décohérence qui fait que les qubits perdent leur état quantique et donc leurs avantages au bout d'un certain temps (de l'ordre de la microseconde pour les qubits supraconducteurs, sinon des dizaines de nanosecondes en général), ou encore le fait que les qubits doivent être isolés de leur environnement pour éviter les erreurs de mesure, ce qui nécessite des températures très basses (moins d'un Kelvin).

Une des implémentations les plus avancées [45] est celle des qubits supraconducteurs [46], ou qubit de charge [47], qui sont des qubits dont l'état est codé dans la charge d'un condensateur

supraconducteur [48, 49].

Un supraconducteur est un matériau qui, en dessous d'une certaine température critique, n'a plus de résistance électrique. La méthode de fabrication des qubits de charge est similaire à celle des autres circuits de microélectronique, en gravant du silicium ou du saphir avec des techniques de lithographie par électrons. Ils utilisent des métaux tel que le niobium ou le tantale.

Le dernier point de construction est les jonctions Josephson [50], qui sont des jonctions entre deux supraconducteurs séparés par une barrière isolante, qui est dans ce cas un oxyde d'aluminium. La méthode de création de ces jonctions est grossièrement la création d'un masque sur le supraconducteur, puis en évaporant le métal à différents angles par le masque.

Le qubit se compose donc d'un condensateur supraconducteur, d'une jonction Josephson et d'une tension qui permet de contrôler la charge du condensateur.

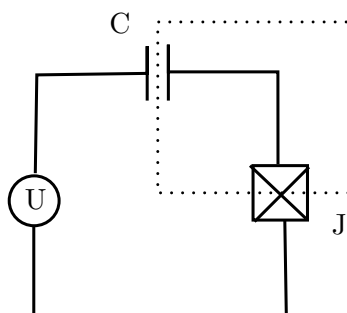


FIGURE 6.2 : Schéma électronique d'un qubit de charge

La région en traitillé représente une “île” de supraconducteur, qui est une région isolée de par le fait qu’une énergie est nécessaire à une charge pour y entrer. Néanmoins, les particules peuvent passer par effet tunnel, qui est dû à la mécanique quantique, et à la nature ondulatoire des particules. De manière classique, une particule ne pourrait pas passer la barrière d’énergie, mais l’équation d’onde de la particule permet de calculer une probabilité de passage, qui est non nulle. Pour les électrons, on arrive à mesurer le passage d’une barrière de quelques nanomètres d’épaisseur, ce qui donne une idée de la précision nécessaire pour la fabrication des qubits.

La jonction Josephson est cette barrière qui va créer un effet tunnel entre les deux supraconducteurs. Les objets quantiques dont on mesure le passage sont des paires de Cooper, qui sont des paires d’électrons liés par une interaction attractive. Celles-ci se forment dans les supraconducteurs, et peuvent être en quelque sorte décrites par une analogie classique. Les électrons sont des particules chargées négativement, et donc se repoussent. Or étant négatif, ils attirent les ions positifs du réseau cristallin du supraconducteur, ce qui crée une déformation du réseau. Cela va créer une densité de charge positive, qui va pouvoir attirer un autre électron, et si l’attraction est plus forte que la répulsion, les deux électrons vont être liés et former une paire de Cooper [51]. Dès lors, le qubit qui représente l’état $|0\rangle$ serait l’absence de paire de Cooper dans l’île et l’état $|1\rangle$ serait la présence de celles-ci.

Pour manipuler ces qubits, que ce soit appliquer une porte ou le mesurer, on va utiliser des micro-ondes, qui sont des ondes électromagnétiques. En envoyant des micro-ondes à une fréquence précise et selon un pattern précis [52], on va pouvoir modifier l’évolution du système, ce qui correspond à ce que l’on a appelé une porte quantique.

Pour la mesure, on va regarder les fréquences de résonance du système, soit les fréquences que le qubit réfléchit qui sont caractéristiques de son état.

On peut également créer une porte contrôlée en envoyant des micro-ondes sur un qubit avec une fréquence de résonance avec un deuxième qubit, ce qui va créer une rotation de l’état

du deuxième qubit autour de l'axe x sur la sphère de Bloch. La direction de la rotation va dépendre de l'état du premier qubit, et donc on a une porte contrôlée. Notons que pour cela, il faut que les deux qubits soient reliés par ce que l'on appelle un bus, qui est grossièrement une cavité faisant résonner les micro-ondes à l'intérieur, et qui relie les deux qubits.

Cette revue de la technologie actuelle est très simplifiée, et ne donne qu'un aperçu de la complexité de la fabrication d'un ordinateur quantique, tant de par la recherche fondamentale nécessaire pour comprendre les phénomènes quantiques, que de par la technologie nécessaire pour manipuler ces phénomènes à grande échelle. Notons que de plus, la puce à proprement parler ne mesure guère plus de quelques centimètres carrés, mais que pour son bon fonctionnement, il faut qu'elle soit proche du zéro absolu pour préserver les propriétés supraconductrices des matériaux, et il faut l'isoler complètement de l'environnement afin d'éviter la décohérence du système, donc il faut créer un vide presque parfait et l'isoler de la lumière et des ondes électromagnétiques entre autres.

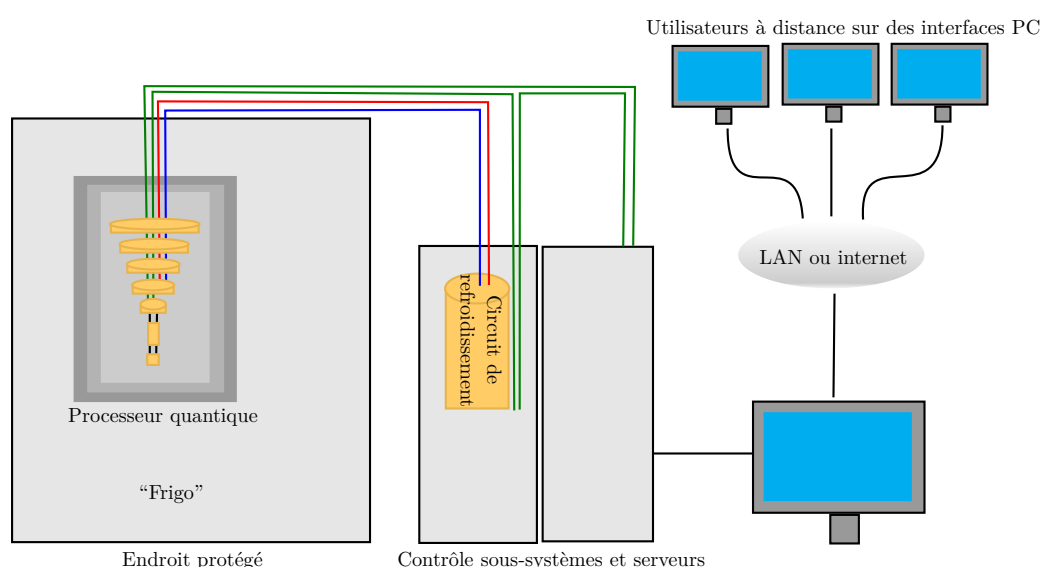


FIGURE 6.3 : Architecture du système D-Wave 2000Q (adapté d'un schéma de la société propriétaire du système)

Sur la figure 6.3, on peut voir une architecture simplifiée d'un ordinateur quantique, avec principalement la zone protégée avec l'ordinateur quantique à proprement parler, sur laquelle est figurée différentes parois de protection en gris, ainsi que la tour de refroidissement en jaune. Celle-ci fait en pratique plus d'un mètre de haut, et tout en bas de celle-ci se trouve la puce. En vert, les câbles de contrôle, qui représente les micro-ondes envoyées sur les qubits pour les manipuler, et en bleu et rouge ceux du système de refroidissement. On y voit également la nécessité de systèmes externes à l'ordinateur quantique afin de l'utiliser, comme un ordinateur classique pour contrôler les micro-ondes, ainsi que des systèmes de refroidissement et de pompage du vide, d'autres de contrôles et de mesures, etc. L'utilisation de ces ordinateurs quantiques se fait donc à distance pour le grand public, si on peut parler de grand public, et cette manière de faire est celle privilégiée par les entreprises qui proposent des ordinateurs quantiques même dans le développement de ceux-ci, car en avoir un chez soi n'est vraisemblablement pas utile, comme on le verra dans les utilisations possibles de ces ordinateurs.

Un ordinateur quantique est un système très complexe, nécessitant une isolation parfaite de l'environnement pour bien fonctionner, et donc il est très difficile de créer un ordinateur quantique avec un grand nombre de qubits. Cela explique aussi qu'actuellement, les ordinateurs

quantiques ne sont pas encore parfaits, mais on peut déjà faire des expériences avec quelques qubits, et espérer que ceux-ci seront de plus en plus performants dans le futur. Dans un ordinateur classique, la logique est réalisée par des transistors dans lesquels passent un courant électrique, alors que dans un ordinateur quantique, on a un système quantique sur lequel on applique des opérations modifiant son état. De part cette différence, le qubit contient son information dans son état, à la différence d'un bit classique qui doit être enregistré dans un système externe pour être conservé.

Troisième partie

Exemples d'algorithmes

Chapitre 7

Algorithme de Deutsch-Jozsa

Premièrement introduit par David Deutsch et Richard Jozsa [53], c'est un des premiers algorithmes quantiques qui montre un avantage exponentiel sur un algorithme classique. Il en demeure un faible intérêt pratique, mais il est important pour comprendre comment exploiter la superposition et l'interférence quantique.

Le problème de Deutsch-Jozsa est une histoire de boîtes noires. On a une boîte noire qui prend en entrée un certain nombre de bits et qui renvoie un bit. On sait que cette boîte noire est soit constante, soit équilibrée, i.e. soit elle renvoie toujours 0, soit elle renvoie 0 pour la moitié des entrées et 1 pour l'autre moitié. Analytiquement, on peut dénoter la boîte noire par une fonction $f : \{0, 1\}^n \rightarrow \{0, 1\}$. Le but de l'algorithme est de déterminer si la boîte noire est constante ou équilibrée.

Commençons par l'algorithme classique. Afin d'être sûr de notre réponse, il faut tester la boîte noire pour la moitié des entrées possibles plus une. En effet, si on a une boîte noire qui renvoie 0 pour la moitié des entrées et 1 pour l'autre moitié, il est possible que les entrées testées soient toutes dans la moitié qui renvoie 0. De fait, tant que cela n'est pas fait, on ne peut pas conclure. Le nombre d'entrées à tester est donc $2^{n-1} + 1$ pour n le nombre d'entrées, ce qui est donc une complexité exponentielle, $\mathcal{O}(2^n)$.

La version quantique de l'algorithme que nous allons présenter n'est pas la version originale de Deutsch et Jozsa, mais une version améliorée par Cleve, Ekert, Macchiavello et Mosca [54].

L'idée de l'algorithme [55] est de mettre tous les qubits en superposition, en notant que celui qui percevra la réponse de la boîte noire doit être initialement à 1. Ensuite, on applique la boîte noire sur tous les qubits. Cela va avoir pour effet de modifier tous les qubits, comme la base n'est plus la base computationnelle, mais la base d'Hadamard. En revenant dans la base computationnelle, via une autre transformation d'Hadamard, car elle est sa propre inverse, la mesure des qubits d'entrée nous donnera la réponse quant à la nature de la boîte noire.

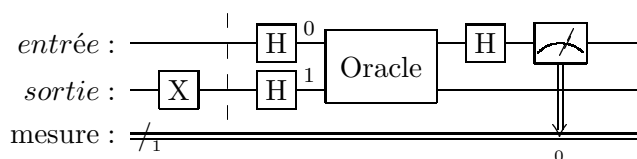


FIGURE 7.1 : Algorithme de Deutsch

La figure 7.1 montre l'algorithme de Deutsch, qui est la version à une entrée de l'algorithme de Deutsch-Jozsa. On considère que l'oracle applique la fonction f sur les qubits d'entrée et renvoie le résultat sur le qubit de sortie sans créer de décohérence.

Pour résoudre le problème de manière classique, on voit qu'il faut appeler la fonction deux fois, une fois avec 0 en entrée et une fois avec 1 en entrée. Dans ce schéma, on voit que l'on appelle la fonction une seule fois, qui est déjà un avantage.

On commence donc après la barrière avec les qubits dans l'état $|0\rangle|1\rangle$, qui deviennent $\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$ après l'application de la première porte d'Hadamard sur chacun des qubits. Ensuite, on applique l'oracle, qui va modifier les qubits d'entrée en fonction de la fonction f comme suit :

$$|x\rangle|y\rangle \rightarrow |x\rangle|f(x) \oplus y\rangle$$

En l'appliquant sur le circuit, on obtient :

$$\frac{1}{2}(|0\rangle(|f(0) \oplus 0\rangle - |f(0) \oplus 1\rangle) + |1\rangle(|f(1) \oplus 0\rangle - |f(1) \oplus 1\rangle))$$

En séparant les cas, on observe que

$$\begin{cases} f(0) = 0 \Rightarrow f(0) \oplus 0 = 0 \text{ et } f(0) \oplus 1 = 1 \Rightarrow (|f(0) \oplus 0\rangle - |f(0) \oplus 1\rangle) = (|0\rangle - |1\rangle) \\ f(0) = 1 \Rightarrow f(0) \oplus 0 = 1 \text{ et } f(0) \oplus 1 = 0 \Rightarrow (|f(0) \oplus 0\rangle - |f(0) \oplus 1\rangle) = (|1\rangle - |0\rangle) = -(|0\rangle - |1\rangle) \end{cases}$$

et similairement pour $f(1)$.

On peut donc réécrire $|f(0) \oplus 0\rangle - |f(0) \oplus 1\rangle$ comme $(-1)^{f(0)}(|0\rangle - |1\rangle)$ et $|f(1) \oplus 0\rangle - |f(1) \oplus 1\rangle$ comme $(-1)^{f(1)}(|0\rangle - |1\rangle)$, ce qui nous donne :

$$\frac{1}{2}((-1)^{f(0)}|0\rangle(|0\rangle - |1\rangle) + (-1)^{f(1)}|1\rangle(|0\rangle - |1\rangle))$$

Mettons ensuite $(-1)^{f(0)}$ en évidence, et comme $(-1)^n$ alterne, on peut écrire $(-1)^{f(1)-f(0)} = (-1)^{f(0) \oplus f(1)}$, ainsi que $(|0\rangle - |1\rangle)$ mis en évidence :

$$(-1)^{f(0)} \frac{1}{2}(|0\rangle + (-1)^{f(0) \oplus f(1)}|1\rangle)(|0\rangle - |1\rangle)$$

L'opérateur \oplus , qui est l'addition modulo 2, est équivalent à l'opérateur XOR, comme évoqué dans la section 3. Or, si $f(0) = f(1)$, l'oracle est constant, ce qui est équivalent à dire que $f(0) \oplus f(1) = 0$. Essayons donc d'extraire cette information du circuit.

Pour la suite on ne se soucie plus du qubit de sortie, car on le voit dans l'équation, ce n'est pas lui qui va nous donner l'information. De plus, la phase globale $(-1)^{f(0)}$ n'est pas observable, car elle ne change pas la probabilité de mesure, on peut donc l'ignorer également. Nous ne soucions donc plus que des qubits d'entrée, et on peut réécrire l'équation comme suit :

$$\frac{1}{\sqrt{2}}(|0\rangle + (-1)^{f(0) \oplus f(1)}|1\rangle)$$

qui devient ensuite par la porte d'Hadamard :

$$\frac{1}{2}(|0\rangle + |1\rangle + (-1)^{f(0) \oplus f(1)}|0\rangle - (-1)^{f(0) \oplus f(1)}|1\rangle) = \frac{1}{2}((1 + (-1)^{f(0) \oplus f(1)})|0\rangle + (1 - (-1)^{f(0) \oplus f(1)})|1\rangle)$$

On constate alors que si $f(0) \oplus f(1) = 0$, alors la probabilité de mesure de $|0\rangle$ est de 1, et si $f(0) \oplus f(1) = 1$, alors la probabilité de mesure de $|1\rangle$ est de 1, ainsi, en mesurant le qubit d'entrée, on peut déterminer si l'oracle est constant ou équilibré.

On peut démontrer le même résultat en prenant n entrées, et en mesurant les n qubits d'entrée, on peut déterminer si l'oracle est constant ou équilibré. Néanmoins, on ne le fera pas ici, car c'est sensiblement la même démonstration à part que l'on traite des sommes sur toutes les

entrées plutôt que des valeurs plus concrètes comme dans ce cas-ci.

Il est amusant de l'essayer avec un circuit réel, comme celui de la figure 7.1, en utilisant comme oracle constant aucune porte, et comme oracle équilibré une porte CX avec comme contrôle l'entrée et comme cible le qubit de sortie. On obtient après exécution de l'algorithme 1024 fois des résultats satisfaisants, soit 51 mesures de 0 pour 973 mesures de 1 pour l'oracle équilibré, et 1017 mesures de 0 pour 7 mesures de 1 pour l'oracle constant. Cela correspond bien à ce que l'on attendait, néanmoins les calculs prédisaient un résultat parfait, ce qui n'est pas le cas ici, mais cela est dû à d'autres facteurs propres aux ordinateurs quantiques, comme les erreurs de portes, les erreurs de mesure, etc.

Ce circuit présente une première idée de ce que permettent la superposition et l'interférence quantique. On voit dans cet algorithme que l'on peut traiter plusieurs cas en même temps, et que l'on peut ensuite extraire l'information de manière constructive. Cela permet aussi de voir que par rapport à un ordinateur classique, on peut avoir un temps constant pour résoudre un problème, alors que l'ordinateur classique a besoin d'un temps exponentiel. De fait, cela peut apparaître comme un avantage non-négligeable sur des problèmes plus complexes.

Chapitre 8

Quelques protocoles

Certains concepts ont déjà été vu plus tôt, néanmoins un certain formalisme est très utile à la bonne compréhension des capacités plus avancées de l'informatique quantique. Ce ne sont pas des algorithmes à proprement parler, mais plutôt des protocoles qui servent de base à la construction d'algorithmes.

8.1 Retour de phase

Ce protocole est très perturbant au premier abord, mais il est en fait très simple. Rappelons l'algorithme de Deutsch-Jozsa vu dans le chapitre du même nom. Lors de l'application de l'oracle, l'opération qui devait être effectuée sur la sortie s'est retrouvée, de par la configuration du circuit, effectuée sur l'entrée.

On définit un *retour de phase* quand le vecteur propre d'une porte sur un qubit est “retourné” sur un autre qubit via une opération de contrôle.

Prenons l'exemple le plus simple, celui d'une porte de CX . Pour rappel, la porte CX est une porte de contrôle qui effectue une porte X sur le qubit cible si le qubit de contrôle est à l'état $|1\rangle$, donc va inverser le qubit cible.

Les figures suivantes indiquent en rouge l'état initial et en vert l'état final. De plus, les qubits de contrôle sont donnés avec un $_c$ et ceux cibles avec un $_t$.

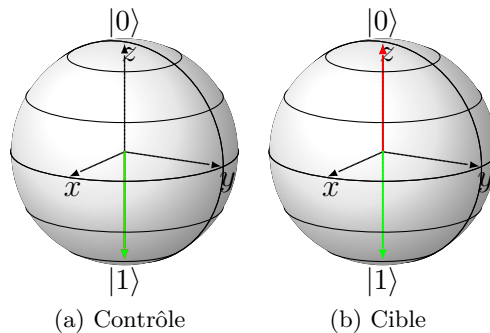


FIGURE 8.1 : Application d'une porte CX sur un qubit à l'état $|1\rangle_c \otimes |0\rangle_t$

Maintenant, remarquons que X aura le même effet sur la base $\{|+\rangle, |-\rangle\}$ que Z sur la base $\{|0\rangle, |1\rangle\}$. De fait, $X|-\rangle = -|-\rangle$, et sachant cela, on peut expérimenter avec une CX appliquée

sur les paires de qubits $|0\rangle_c \otimes |-\rangle_t$ et $|1\rangle_c \otimes |-\rangle_t$:

$$\begin{aligned} |0\rangle_c \otimes |-\rangle_t &\xrightarrow{CX} |0\rangle_c \otimes |-\rangle_t \\ |1\rangle_c \otimes |-\rangle_t &\xrightarrow{CX} |1\rangle_c \otimes X|-\rangle_t = -|1\rangle_c \otimes |-\rangle_t \end{aligned}$$

Dans ces cas-là, seule la phase globale est affectée, et ce n'est donc pas observable, donc peu intéressant. Néanmoins, si le qubit de contrôle est en superposition, alors on peut observer des phénomènes intéressants, comme avec l'application d'une porte CX sur l'état $|+\rangle_c \otimes |-\rangle_t$:

$$|+\rangle_c \otimes |-\rangle_t \xrightarrow{CX} \frac{1}{\sqrt{2}} (|0\rangle_c \otimes |-\rangle_t - |1\rangle_c \otimes |-\rangle_t)$$

que l'on peut séparer de manière équivalente en :

$$\frac{1}{\sqrt{2}} (|0\rangle_c \otimes |-\rangle_t - |1\rangle_c \otimes |-\rangle_t) = \frac{1}{\sqrt{2}} (|0\rangle_c - |1\rangle_c) \otimes |-\rangle_t = |-\rangle_c \otimes |-\rangle_t$$

La vision de cette opération sur la sphère de Bloch est très instructive.

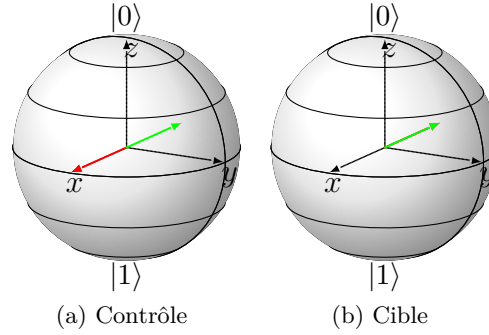
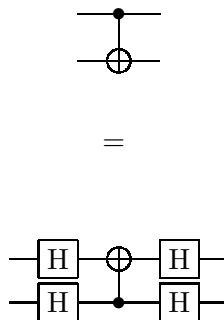


FIGURE 8.2 : Application d'une porte CX sur un qubit à l'état $|+\rangle_c \otimes |-\rangle_t$

En comparant la figure 8.2 avec la figure 8.1, on remarque que l'application d'une porte CX sur un qubit à l'état $|+\rangle_c \otimes |-\rangle_t$ va modifier l'état du qubit de contrôle, à l'inverse de ce qui se passe avec dans la base standard $\{|0\rangle, |1\rangle\}$.

Cet exemple répond complètement à la définition : via la porte de contrôle CX , l'opération, on arrive en changeant de base à modifier l'état du qubit de contrôle. De plus, ce changement de base est aisé, se faisant simplement en appliquant une porte H sur les qubits désirés, ainsi que réversible. Il en découle l'identité décrite dans la figure 8.3. Celle-ci permet de choisir librement le contrôle et la cible d'une porte CX , dans le cas où l'architecture ne le permet pas directement et cela n'est qu'un exemple parmi tant d'autres.

FIGURE 8.3 : Changement de sens d'une porte CX en changeant de base

Le retour de phase est donc un protocole très utile, car il implique que selon l'état des qubits, une opération contrôlée peut être effectuée dans un sens ou dans l'autre, et cela peut être très utile dans la construction de circuits.

8.2 Transformée de Fourier quantique

Les séries de Fourier [56] sont des outils mathématiques très puissants, permettant de décomposer une fonction périodique en une somme de fonctions sinusoïdales. Ces séries sont très utilisées en traitement du signal, en analyse de données, et de nombreuses autres applications.

En effet, elles permettent de décomposer un signal en une somme de signaux plus simples, et donc de simplifier l'analyse de celui-ci. L'idée des séries de Fourier est de décomposer une fonction périodique f en une somme de fonctions sinusoïdales de fréquences différentes, et en généralisant aux nombres complexes, elle s'exprime via la formule d'Euler :

$$e^{i\theta} = \cos(\theta) + i \sin(\theta)$$

On appelle transformée de Fourier la transformation qui permet de passer d'une fonction f à sa décomposition en série de Fourier, généralisée à toute fonction, même non périodique. Dans le cas qui nous intéresse, seule la version discrète [57] de la transformée de Fourier nous intéresse, et elle associe un vecteur (x_0, \dots, x_{N-1}) à un vecteur (y_0, \dots, y_{N-1}) tel que :

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i \frac{jk}{N}}$$

et similairement de manière quantique [58], on associe un état $|X\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$ à un état $|Y\rangle = \sum_{k=0}^{N-1} y_k |k\rangle$ selon la même formule.

Elle fait ainsi la transformation entre deux bases, celle Z et celle de Fourier. La visualisation intuitive de celle-ci est suffisante pour comprendre son fonctionnement et son implémentation.

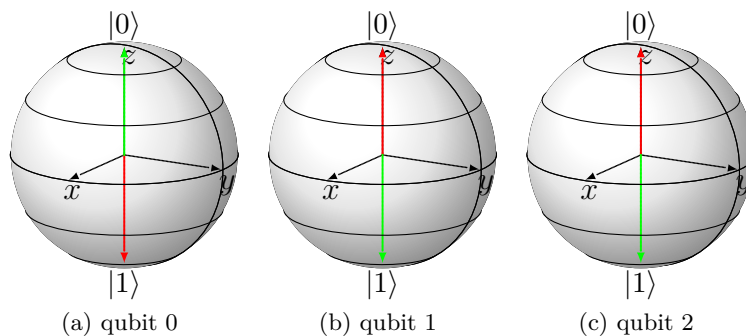
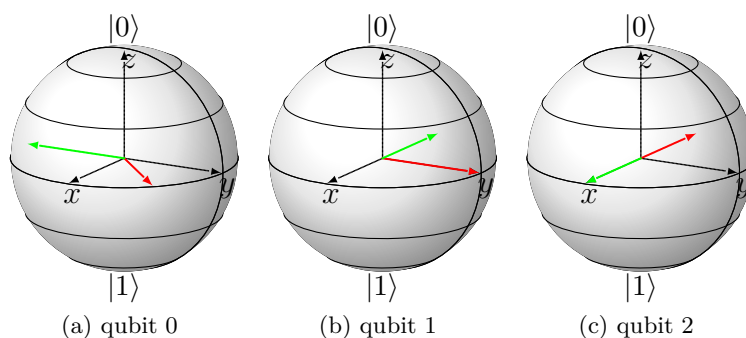
FIGURE 8.4 : Nombres dans la base Z (1 en rouge, 6 en vert)

FIGURE 8.5 : Nombres dans la base de Fourier (1 en rouge, 6 en vert)

Dans la figure 8.4, les nombres sont en binaire (i.e. 1 correspond à 100 et 6 à 011) en utilisant la base $|0\rangle$ et $|1\rangle$, tandis que dans la figure 8.5, on observe que les nombres sont encodés dans la rotation autour de l'axe Z . On note ce nouvel état $QFT|x\rangle = |\tilde{x}\rangle$.

Comme on voit sur la figure 8.5, le nombre 6 est encodé sur trois qubits via une rotation de $\frac{6}{2^n} = \frac{6}{2^3} = \frac{3}{4}$ d'un tour complet sur le qubit 0, $\frac{3}{2}$ d'un tour complet sur le qubit 1 et une fois encore, on double cela pour le qubit 2.

Nous ne nous attarderons pas sur l'implémentation plus que cela, car elle fait exactement ce que l'on vient de décrire, soit l'application d'une porte d'Hadamard, puis la rotation autour de l'axe Z selon la valeur des autres qubits via des portes de phase contrôlées, ensuite l'opération se répète sur les qubits suivants.

Néanmoins, on se rend compte que le circuit grossit assez vite, et à un certain stade, pour économiser des ressources, on peut ignorer certaines rotations tout en gardant une précision suffisante.

Dernier point, comme toutes les opérations, celle-ci est réversible, et on peut définir l'opération inverse comme QFT^\dagger .

La transformée de Fourier quantique permet donc de profiter des différentes bases offertes par les qubits, par exemple dans ce cas en encodant les nombres différemment. Dans ce cas-là, cela va même plus loin, car cela n'est pas comme en informatique classique où l'on se base sur deux états, mais on exploite une propriété quantique qui est la phase.

8.3 Estimation de phase quantique

Tout comme la section précédente, pour ce protocole, nous laisserons quelque peu de côté la partie mathématique, car l'aspect intuitif est bien plus intéressant et pertinent.

Soit un opérateur U et un état $|\psi\rangle$ tel que $U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle$, l'algorithme d'estimation de phase quantique [59] permet d'estimer θ . On note donc que $|\psi\rangle$ est un vecteur propre de U avec la valeur propre $e^{2\pi i\theta}$.

Pour ce faire, on initialise un registre de t qubits à $|0\rangle$, et un autre registre d'un qubit à $|\psi\rangle$. Ensuite, on fait passer le registre de t qubits à travers une porte d'Hadamard, puis le but est d'encoder la phase de U dans la base de Fourier sur le registre. Pour ce faire, on utilise le retour de phase d'une porte CU , avec un des qubits du registre de t qubits comme contrôle et $|\psi\rangle$ en cible. Sur le premier qubit du registre de t qubits, la porte CU 2^{t-1} fois, le second 2^{t-2} fois, et ainsi de suite jusqu'au dernier qubit où la porte CU est appliquée une fois.

De là, on applique la transformée de Fourier inverse sur le registre de t qubits, puis on mesure le registre. La mesure de celle-ci correspond à $2^t\theta$, et dans ce cas si c'est un nombre entier, on aura la solution exacte, sinon, on aura une probabilité bien plus élevée pour les valeurs autour de la solution.

Si l'on construit le circuit avec un angle $\theta = \frac{1}{3}$, on utilisera pour l'exemple la porte $P(\frac{2\pi}{3})$, et on note que $P(\frac{2\pi}{3})|1\rangle = e^{2\pi i\frac{1}{3}}|1\rangle$, donc $|1\rangle$ est un vecteur propre de notre opérateur.

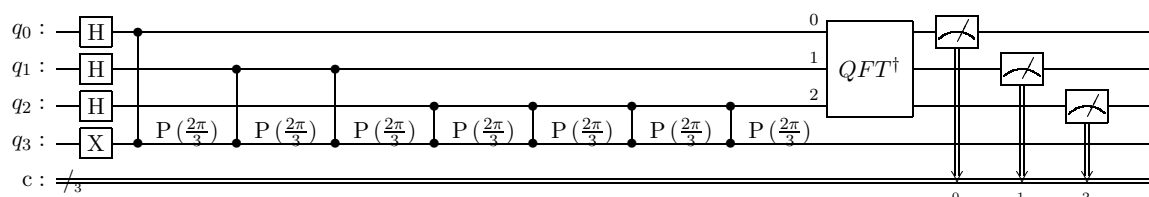
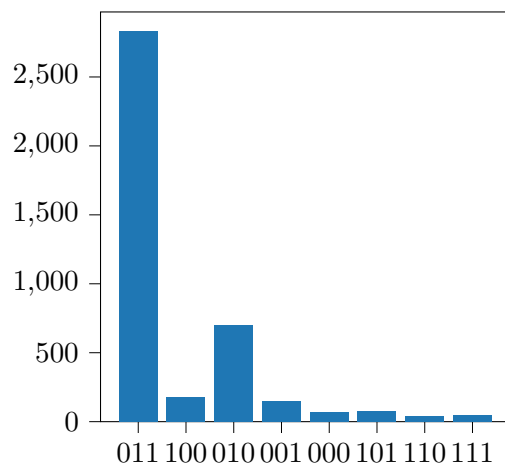


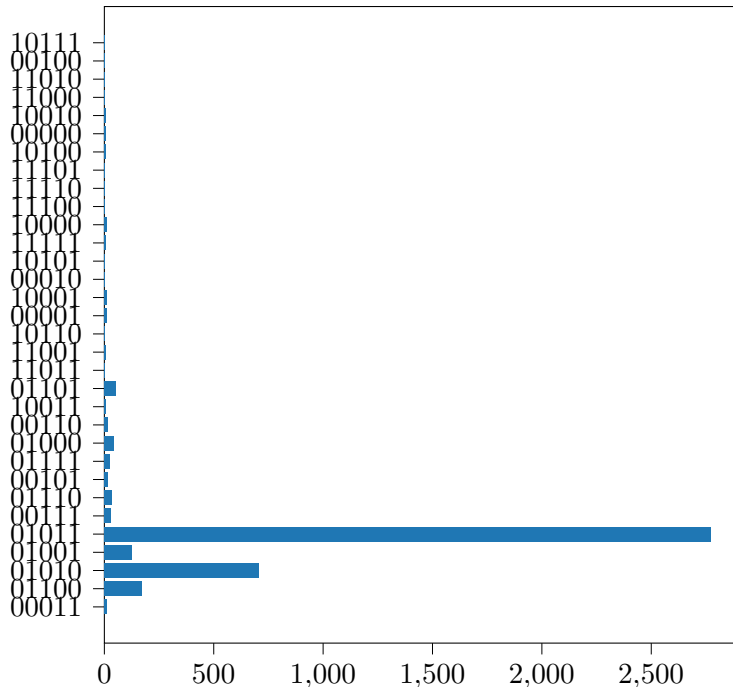
FIGURE 8.6 : Estimation de phase pour $\theta = \frac{1}{3}$ avec un registre de $t = 3$ qubits

Les qubits 0 à 2 sont le registre de t qubits, le qubit 3 est le registre de $|\psi\rangle$, que l'on initialise selon le protocole énoncé précédemment. Petite subtilité, par rapport à ce qui a été dit auparavant, on applique la porte CU le plus de fois sur le dernier qubit du registre de t qubits, et non le premier. Si l'on ne le faisait pas, les résultats obtenus seraient inversés, et serait donc moins lisibles, cela est dû à la manière dont le module utilisé pour la simulation des circuits quantiques et l'interaction avec les ordinateurs à disposition fonctionne.

FIGURE 8.7 : Mesures pour $\theta = \frac{1}{3}$ avec un registre de $t = 3$ qubits

Les deux plus grandes valeurs obtenues sont 011 qui correspond à 3 et donc un angle de $\theta \approx \frac{3}{2^3} = 0.375$ et 010 qui correspond à 2 et donc un angle $\theta \approx \frac{2}{2^3} = 0.25$. Elles entourent bien la valeur de θ que l'on cherchait à estimer, néanmoins la précision laisse à désirer.

Pour l'augmenter, on peut augmenter le nombre de qubits du registre de t qubits. Regardons par exemple avec 5 qubits. Outre le gigantisme du circuit 8.8, problème qui sera discuté ultérieurement avec des pistes d'amélioration, on peut remarquer que les mesures sont bien plus précises que dans le cas précédent. On obtient principalement une mesure de 01011, soit 11 en décimal, ce qui donne une estimation de $\frac{11}{32} \approx 0.344$.

FIGURE 8.9 : Mesures pour $\theta = \frac{1}{3}$ avec un registre de $t = 5$ qubits

On a vu avec la transformée de Fourier quantique que la phase offrait une nouvelle dimension pour l'encodage des nombres. La phase peut offrir d'autres possibilités, et donc la possibilité de mesurer cette propriété est intéressante. Il faut préciser que la transformée de Fourier quantique

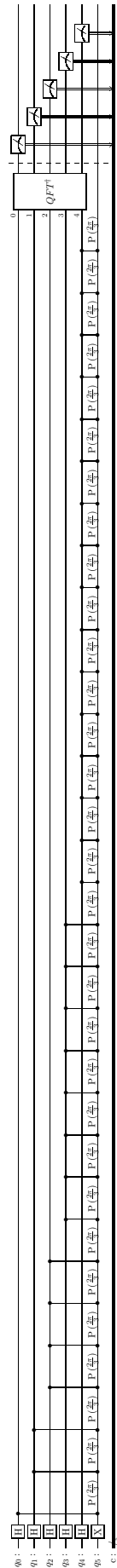


FIGURE 8.8 : Estimation de phase pour $\theta = \frac{1}{3}$ avec un registre de $t = 5$ qubits

se base sur la phase des qubits, alors que l'estimation de phase quantique mesure la phase qu'induit un opérateur sur un état propre de celui-ci. Néanmoins, l'un a l'autre peuvent être relié, car on peut encoder dans la base de Fourier par un opérateur et on peut mesurer la phase de celui-ci. De fait, les deux notions sont assez liées dans ce sens-là, même si elles peuvent être utilisées indépendamment l'une de l'autre. De plus, cela revêt un aspect aussi conceptuel, car par cette mesure, on donne une valeur concrète à une propriété quantique, basée sur les nombres complexes qui plus est.

Chapitre 9

Algorithme de Shor

L'algorithme de Shor fut développé en 1994 par Peter Shor [60]. Il permet de factoriser un nombre entier N en un produit de deux nombres premiers avec une complexité temporelle en $\mathcal{O}((\log N)^3)$ et spatiale en $\mathcal{O}(\log N)$. En comparaison, de manière classique, on est supérieur à $\mathcal{O}(N^k)$, plus autour de $\mathcal{O}(e^N)$ [61]. L'intérêt de cet algorithme est donc évident pour la cryptographie, qui s'appuie sur cette complexité afin de garantir la sécurité des communications, via des codes comme RSA. Néanmoins, son impact est quand même à modérer, car même s'il revenait à être implémenté, il existe d'autres méthodes de cryptographie dite post-quantiques, qui ne sont pas sensibles à ces découvertes.

9.1 Principe

L'algorithme de Shor [62] se base sur une partie classique, et une partie quantique. La partie classique est la génération d'un nombre aléatoire a inférieur à N . Ensuite calculer le plus grand diviseur commun de a et N , si celui-ci est différent de 1, alors on a trouvé un facteur de N . Sinon, on passe à la partie quantique. Le calcul du PGDC peut se faire via l'algorithme d'Euclide par exemple.

Algorithme 6 : Algorithm d'Euclide

Entrées : Deux entiers a et b

Sorties : Le PGDC de a et b

tant que $a \neq b$ **faire**

si $a > b$ **alors**

$a \leftarrow a - b$;

sinon

$b \leftarrow b - a$;

fin

fin

retourner a ;

La partie quantique vise à calculer la période d'une fonction $f(x) = a^x \bmod N$, autrement dit, le plus petit entier r tel que $f(x+r) = f(x)$. Si r est impair, ou que $a^{r/2} \bmod N = -1$, alors on recommence avec un autre a . Sinon, on peut tirer les conclusions suivantes :

1. Comme pour $x = 0$, $f(0) = a^0 \bmod N = 1$ donc $f(0+r) = f(r) = 1$, on obtient $a^r \bmod N = 1$
2. Cela implique $a^r - 1 \bmod N = 0$, ainsi N divise $a^r - 1$.

3. Comme r est pair, on peut écrire $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$.
4. Comme r est la plus petite période, $a^{r/2} - 1$ n'est pas divisible par N . On peut donc calculer le PGDC de $a^{r/2} - 1$ et N afin de trouver un facteur de N . Si celui-ci vaut 1, alors on recommence avec un autre a , et sinon, on a réussi.

Il a été prouvé que cette méthode devrait fonctionner après quelques essais.

La partie quantique est principalement une estimation de phase, voir 8.3. L'astuce consiste à utiliser un opérateur U tel que $U|x\rangle = |ax \bmod N\rangle$. Dans ce cas, nous avons $U^n|x\rangle = |a^n x \bmod N\rangle$.

Deux problèmes se posent alors, premièrement, il faut un vecteur propre de U afin de pouvoir utiliser l'algorithme d'estimation de phase, et deuxièmement, il faut pouvoir implémenter U . Le premier problème peut être résolu en définissant des états comme suit :

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-\frac{2\pi i s k}{r}} |a^k \bmod N\rangle$$

qui sont des vecteurs propres de U associés à la valeur propre $e^{\frac{2\pi i s}{r}}$. On peut s'en convaincre car dans l'anneau \mathbb{Z}_N , on a $a^r = a^0 = 1$ et appliquer $U|u_s\rangle$ revient globalement à faire la somme de $k = 1$ à r mais par la remarque précédente, c'est équivalent à faire la somme de $k = 0$ à $r - 1$. De plus, la somme de vecteurs propres de U est aussi un vecteur propre de U . Si l'on choisit de faire

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle$$

car les différentes phases s'annulent. De fait, nous avons un vecteur propre de U .

Notons via cela que la phase obtenue sera $\theta = \frac{s}{r}$, et que pour déduire r il faudra transformer la phase en une fraction avec le dénominateur inférieur à N , et celui-ci sera r .

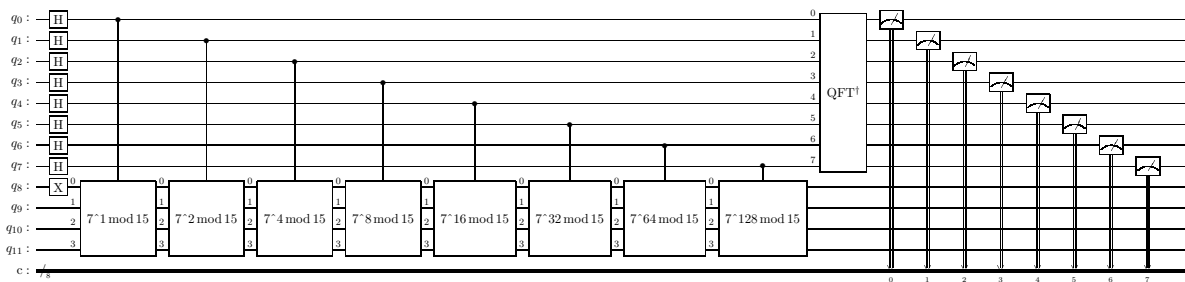
Le second problème est de construire U . Pour ce faire on peut implémenter l'algorithme classique pour l'exponentiation modulaire, de complexité polynomiale, néanmoins comme la construction concrète n'est pas évidente et demeure la raison principale de ralentissement de l'algorithme, nous ne la détaillerons pas ici. Nous utiliserons celle construite dans le module `Qiskit`. Notons que nous utilisons à chaque fois des portes U^{2^j} , sachant que si $a \bmod N = b$, alors $a^c \bmod N = b^c$.

Algorithme 7 : Exponentiation modulaire

Entrées : Trois entiers a , j et N
Sorties : $a^{2^j} \bmod N$
pour $i \leftarrow 1$ **à** j **faire**
 | $a \leftarrow a^2 \bmod N$;
fin
retourner a ;

9.2 Implémentation simple

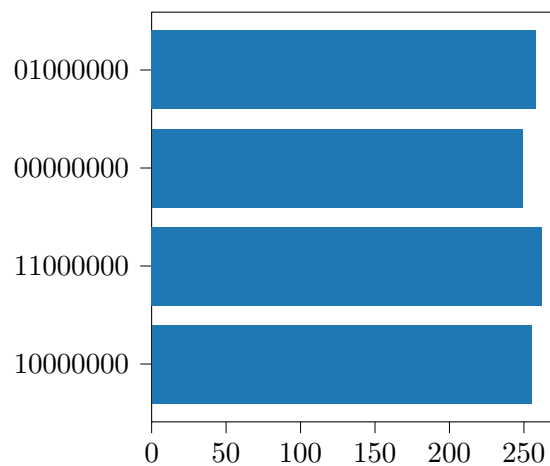
Pour montrer le fonctionnement de l'algorithme, nous allons implémenter une version simple de celui-ci, afin de factoriser $N = 15$ à partir de $a = 7$.

FIGURE 9.1 : Algorithmes de Shor pour $N = 15$ et $a = 7$

Nous distinguons deux parties au circuit. La première partie, en haut, est le même que pour l'estimation de phase, la partie avec la mesure. La seconde partie, en bas, est initialisée dans l'état $|1\rangle$, et est composée de l'application successive des portes U , et notons également que nous utilisons dans ce cas une porte U spécialisée pour $a = 7$ et $N = 15$, néanmoins, il est possible de construire une porte U pour n'importe quel a et N en prenant quelques considérations supplémentaires (principalement la taille des deux nombres).

9.2.1 Simulation

La simulation du circuit fait apparaître, dans ce cas, quatre valeurs possibles pour la mesure de la partie supérieure du circuit.

FIGURE 9.2 : Simulation du circuit de Shor pour $N = 15$ et $a = 7$

En convertissant en décimal, nous obtenons les valeurs 0, 64, 128 et 192. En divisant par $2^8 = 256$, on obtient des phases de 0, 0.25, 0.5 et 0.75 qui correspondent aux fractions $\frac{0}{1}$, $\frac{1}{4}$, $\frac{1}{2}$ et $\frac{3}{4}$. De fait, deux mesures nous donnent la bonne valeur de $r = 4$, car l'algorithme de Shor peut échouer si $s = 0$ ou s et r ne sont pas premiers entre eux, où il faut alors essayer d'amplifier les fractions également.

En effet, $a^{r/2} - 1 = 7^{4/2} - 1 = 49 - 1 = 48$ et ensuite le PGDC de 48 et 15 est 3, qui est un facteur de 15.

9.2.2 Hardware réel

Nous avons également exécuté le circuit sur un hardware réel, en prenant , et nous avons obtenu les résultats suivants.

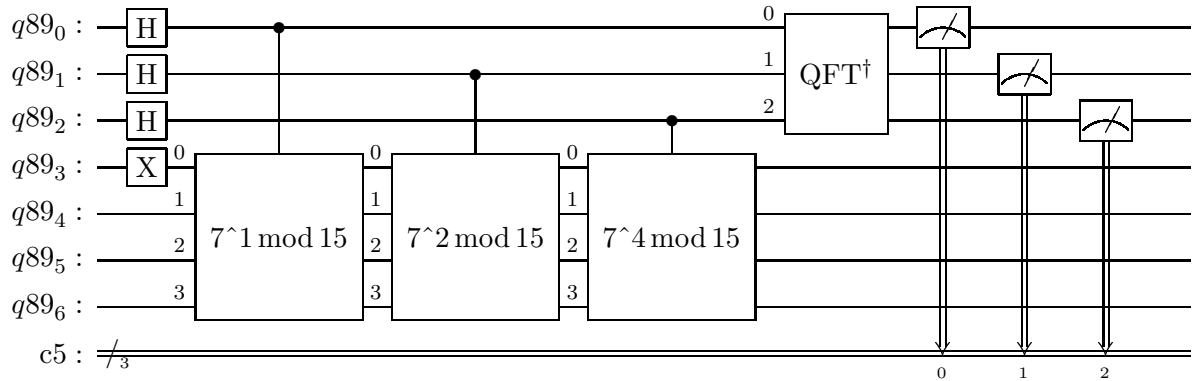


FIGURE 9.3 : Algorithmes de Shor pour $N = 15$ et $a = 7$, version à 3 qubits pour encoder l'angle

Afin de donner une idée de la qualité des résultats, nous avons également appliqué une réduction d'erreur automatique, calibrée pour le hardware utilisé.

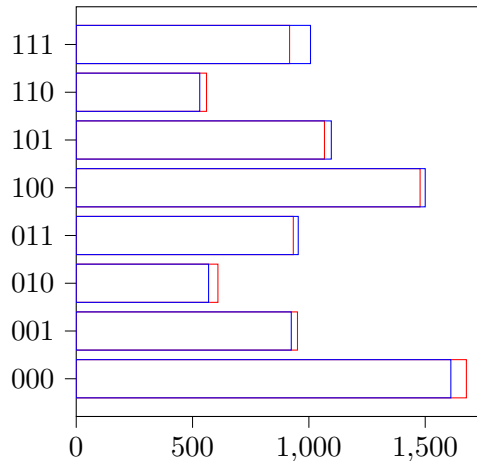


FIGURE 9.4 : Execution du circuit de Shor pour $N = 15$ et $a = 7$ sur un hardware réel, avec en rouge les valeurs brutes, et en bleu après une atténuation d'erreur ¹

Sur les machines à dispositions, nous voyons que les résultats sont peu encourageants, car outre la solution triviale 000, celle qui ressort le plus est 100, qui correspond à $r = 2$, parce que cela revient à $\frac{4}{2^3} = \frac{1}{2}$, hors comme dit plus haut ce n'est pas la bonne solution, car $7^2 \bmod 15 = 4$ alors que l'on recherche $7^r \bmod 15 = 1$. Les mesures pour $r = 4$ ne ressortent pas à cause du bruit, néanmoins la solution $\theta = \frac{1}{2}$ est liée au fait que $s = 2$, et donc que cela se simplifie en $\frac{2}{4} = \frac{1}{2}$.

¹Exécuté le 31.08.2023 sur la machine 'ibm_nairobi', job id : cjo1svhpthn588jkkphg ; avec pour la réduction d'erreur automatique, job id : cjo1ngpdll3gjfa1lne0

Une explication à l'apparition des mesures de type b_1b_21 , qui sont forcément fausses, car cela donnerait une fraction au dénominateur de 2^3 , qui est supérieur au r recherché. Cela est dû à l'erreur sur la porte U^4 , qui est la porte la plus grande et donc avec le plus d'erreur. Or dans les bons résultats, ce bit est nul, ce qui nous indique qu'en simulant le circuit cette porte devrait avoir aucun effet, mais en l'exécutant sur un hardware réel, cela n'est pas le cas, et donc cela crée de l'erreur de manière non négligeable, comme on le voit en comparant les résultats 9.4 et 9.5.

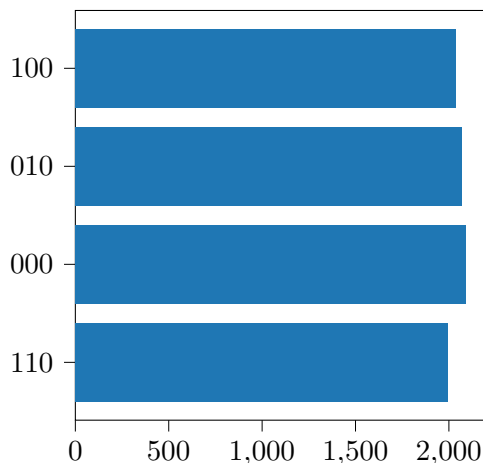


FIGURE 9.5 : Simulation du circuit de Shor pour $N = 15$ et $a = 7$ utilisé sur un vrai ordinateur

Sur ce dernier diagramme, on observe néanmoins effectivement les mesures 110 et 010 qui donnent après traitement $r = 4$, et donc qui sont les bonnes mesures.

9.3 Application à un problème concret

La factorisation d'un nombre en un produit de deux nombres premiers est d'une importance majeure en cryptographie. La plupart des systèmes de cryptographie actuels reposent sur la difficulté de factoriser les grands nombres, tel que le célèbre RSA.

Le système RSA fonctionne comme suit [63]. Prenons deux personnes Alice et Bob. Alice crée les clés de chiffrement en choisissant deux nombres premiers p et q distincts. Alors, elle calcule le nombre $n = pq$, puis $\phi(n) = (p - 1)(q - 1)$, où ϕ est la fonction qui retourne la valeur de l'indicatrice d'Euler en n , soit le nombre de nombres premiers avec n inférieurs à n . Cette fonction est multiplicative, c'est-à-dire que $\phi(uv) = \phi(u)\phi(v)$, et si p est premier, alors $\phi(p) = p - 1$. Après cela, Alice choisit un nombre e premier avec $\phi(n)$, et calcule d tel que $ed \bmod \phi(n) = 1$.

Alors la clé publique est le couple (n, e) , et la clé privée est d . Pour chiffrer un message M inférieur à n , Bob calcule $C = M^e \bmod n$, et pour déchiffrer, Alice calcule $M = C^d \bmod n$. Cela se démontre en utilisant le petit théorème de Fermat, qui dit que si p est premier, $M^{p-1} \bmod p = 1$. Alors, on a $C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$. Or $ed \bmod (p - 1)(q - 1) = 1$, ce qui est équivalent à $ed = 1 + k(p - 1)(q - 1)$ pour un certain $k \in \mathbb{N}$. De fait, d'après le petit théorème de Fermat, $M^{ed} \bmod p = M^{1+k(p-1)(q-1)} \bmod p = M \bmod p$ et de même $M^{ed} \bmod q = M \bmod q$. De fait, $M^{ed} - M$ est congru à 0 modulo p et q , et donc divisible par p et q , et comme p et q sont premiers entre eux, $M^{ed} - M$ est divisible par $pq = n$, ce qui implique que $M^{ed} \bmod n = M$. Le système repose donc sur la difficulté de factoriser n en p et q , et donc de calculer $\phi(n)$.

Prenons un exemple concret. Issu du cours de mathématiques appliquées de deuxième année de monsieur Klose, gymnase Auguste Piccard.

On donne le couple $(E; n) = (17; 143)$, ainsi que le message chiffré (en blocs)

14; 112; 49; 117; 17; 81; 53; 37; 49

Déchiffrez ce message.

Ce problème peut être pris comme un exemple réduit de quelqu'un voulant craquer un message en connaissant la clé publique.

On va donc construire un circuit qui va calculer la factorisation de $N = n = 143$ via l'algorithme de Shor. En générant $a = 12$, via le constructeur de circuits de Shor proposé par Qiskit [64, 65], on obtient le circuit suivant.

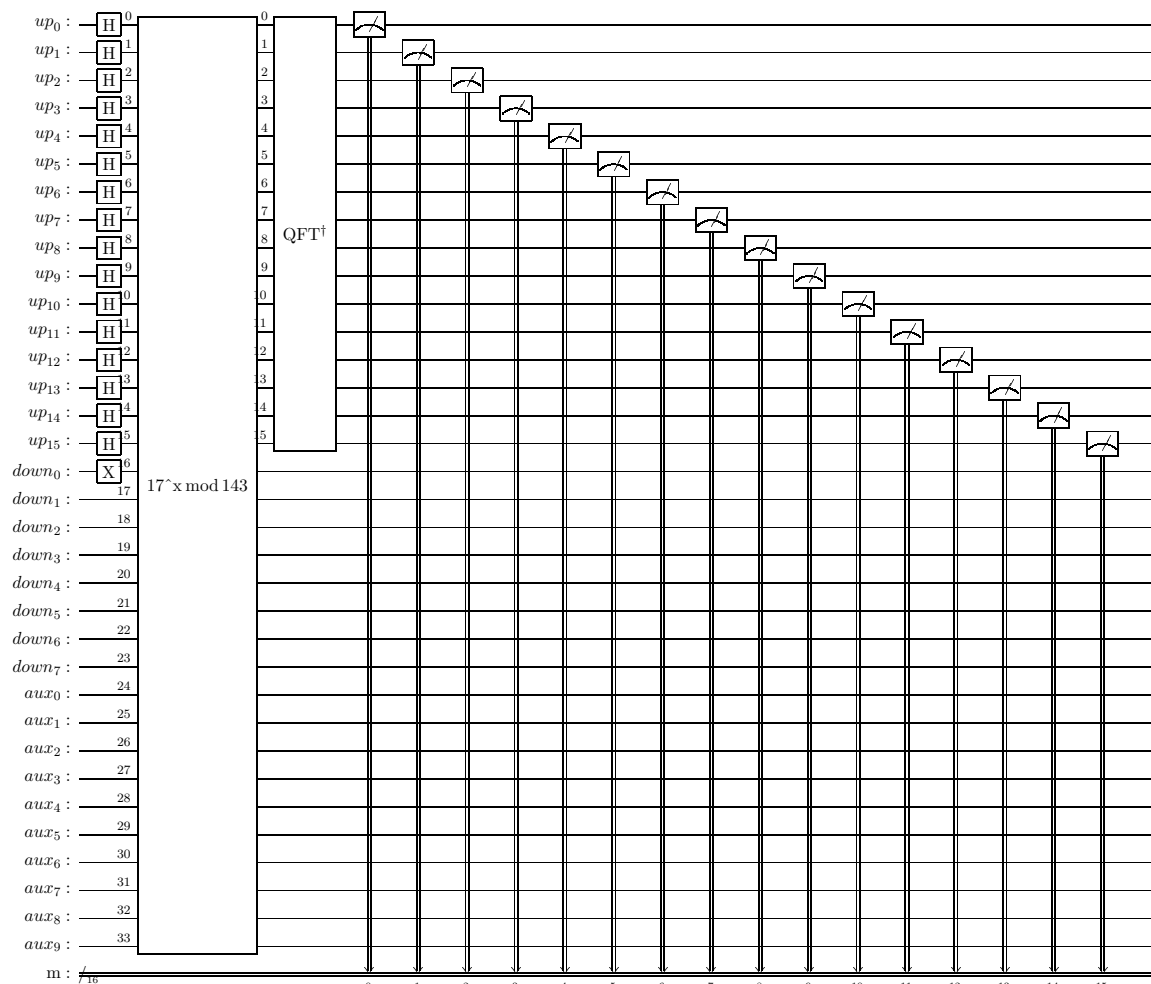


FIGURE 9.6 : Algorithmes de Shor pour $N = 143$ et $a = 12$

Un tel circuit étant trop gros pour être simulé par un ordinateur privé, on compte sur le simulateur par produit matriciel d'IBM. On obtient alors la sortie suivante :

The list of factors of 143 as computed by the Shor's algorithm is [11, 13]

qui est bien la factorisation de 143.

Notons donc plusieurs choses. Tout d'abord la difficulté de simulation de tels circuits. En effet, on considère ici un circuit de taille 34, or le maximum envisageable pour un ordinateur de tout un chacun est de l'ordre de 32, et la ressource nécessaire pour le simuler est exponentielle en la taille du circuit. De plus, l'exécution de ce circuit sur un ordinateur quantique est difficilement envisageable car il nécessite des ordinateurs réservés à la recherche. Ensuite, on peut noter l'utilisation de $a = 12$ qui est un choix pratique, car d'autres essais avec par exemple $a = 2$ ou $a = 17$ ne permettent pas d'obtenir la factorisation de N .

Après avoir obtenu la factorisation de 143, il est aisé de finir le problème proposé plus haut. On commence par calculer $\phi(143)$, qui vaut $(11 - 1)(13 - 1) = 120$. Puis on cherche $17 \cdot d \bmod 120 = 1$, qui peut se calculer par exemple en utilisant le fait que $a^{\phi(n)} \bmod n = 1$ et donc l'inverse est $a^{-1} \bmod n = a^{\phi(n)-1} \bmod n$, ce qui donne $17^{119} \bmod 120 = 113 = d$. En utilisant la méthode de décodage, nous obtenons alors le message suivant :

27; 8; 4; 13; 62; 9; 14; 20; 4

qui correspond selon la table alphabétique du cours en question à :

Bien joue

et donc le message est bien décodé.

Par là, on voit un avantage possible de l'ordinateur quantique, déjà dans la réalisation de calculs avec une complexité moindre qu'un ordinateur classique, mais aussi dans la possibilité de travailler en complémentarité avec un ordinateur classique, en utilisant l'ordinateur quantique pour des calculs spécifiques, et l'ordinateur classique pour des calculs plus simples, dans ce cas toute la préparation de l'algorithme par ordinateur classique, et la recherche de la période par ordinateur quantique. De plus, on entre dans un des domaines dans lequel l'ordinateur quantique est le plus étudié, la cryptographie, et dans ce cas la remise en question de nos méthodes de chiffrement actuelles, qui se base sur la complexité nécessaire à factoriser un nombre de manière classique, et qui serait mis en danger par un ordinateur quantique.

Chapitre 10

Cryptographie : distribution de clés

On a vu dans le chapitre précédent que certains systèmes à clé publique seraient mis en danger par l'existence d'un ordinateur quantique. Cependant, il existe aussi des avantages à utiliser les propriétés quantiques de la matière pour sécuriser des communications.

Par exemple, on peut utiliser la polarisation de photons pour envoyer des qubits d'information via des fibres optiques, ce qui ferait un canal de communication quantique. Cela peut être utilisé pour distribuer des clés de façon sécurisée, c'est-à-dire sans que quelqu'un puisse écouter la communication et que cela soit détecté, ce qui est impossible avec des moyens classiques. Ces clés seront ensuite secrètes et pourront être utilisées pour chiffrer des messages avec un algorithme symétrique classique.

On va prendre nos acteurs habituels pour expliquer le principe de la distribution de clés quantique. Donc Alice et Bob veulent se partager une clé secrète, et Eve veut écouter la communication. Le protocole compte sur le fait que la mesure d'un qubit modifie son état, et donc que si Eve écoute la communication, Alice et Bob le sauront.

Avec le circuit ci-dessous, on peut voir que sans interception, Alice met son qubit dans un état superposé, et Bob le remet dans la base standard avant de le mesurer.

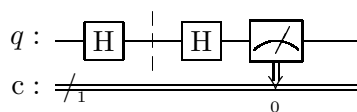


FIGURE 10.1 : Sans interception (Alice | Bob)

Mais si Eve intercepte le qubit, elle va effectuer une mesure sur le qubit, ce qui va le faire passer dans la base standard, et donc Bob va le remettre en superposition avant la mesure et le résultat sera aléatoire.

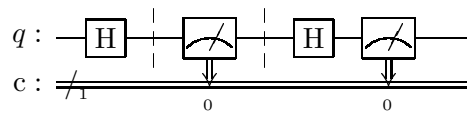


FIGURE 10.2 : Avec interception (Alice | Eve | Bob)

Sachant cela, on peut établir un protocole de distribution de clés quantique comme suit :

1. Alice choisit une liste de bits aléatoires (par exemple : 01001), et pour chaque bit, elle choisit une base de mesure aléatoire (standard ou Hadamard, p.ex. : ZZXXZ). Elle garde ces informations pour elle.
2. Alice encode ensuite chaque bit de sa liste dans un qubit, en utilisant la base correspondante (ce qui donne avec les exemples précédant : $|0\rangle |1\rangle |+\rangle |+\rangle |1\rangle$). Elle envoie ensuite les qubits à Bob.
3. Bob mesure alors chaque qubit aléatoirement dans une base standard ou Hadamard (p.ex. : ZXZXZ). Il garde les résultats pour lui.
4. Bob et Alice partagent alors les bases utilisées pour chaque qubit, et jettent les qubits pour lesquels ils n'ont pas utilisé la même base. Ceux pour lesquels ils ont utilisé la même base sont alors utilisés pour générer la clé (dans notre exemple : 001).
5. Pour vérifier que personne n'a écouté la communication, Alice et Bob prennent une partie de la clé, et la comparent. Si elles sont identiques, alors personne n'a écouté la communication. Sinon, ils recommencent le protocole. En pratique, c'est un poil plus complexe, car il faut prendre en compte le fait que les qubits peuvent être différents simplement à cause du bruit, et en pratique, on doit le prendre en compte dans le partage de la clé et dans la vérification.

On voit donc par cela que l'utilisation de canaux de communication quantique permettrait de distribuer des clés de façon sécurisée, et donc de chiffrer des messages de manière plus sûre. Les technologies quantiques ne se limitent donc pas à la puissance de calcul, mais peuvent aussi être utilisées dans de nombreux autres domaines, comme les communications.

Chapitre 11

Algorithme de Grover

Un autre domaine d'intérêts de l'ordinateur quantique serait dans de l'organisation logistique. L'algorithme de Grover [66, 67] en est un bon exemple, car il permet de trouver un élément dans une base de données non triée. Comme discuté en introduction, la complexité de manière classique est $\mathcal{O}(n)$ dans le cas d'une base de données non structurée, alors que l'algorithme de Grover permet de trouver l'élément en $\mathcal{O}(\sqrt{n})$.

L'exemple type de problème de recherche dans une base de données est le problème de la recherche d'un numéro de téléphone dans un annuaire téléphonique. En effet, si l'annuaire référence les noms de manière alphabétique, trouver un numéro de téléphone revient à rechercher quelque chose sans pouvoir se situer dans l'annuaire.

De plus, l'algorithme de Grover permet d'accélérer la recherche d'une solution à un problème dont on sait vérifier la validité d'une solution, ceux de classe NP . Malgré cela, un problème de classe NP est généralement exponentiel avec le nombre d'éléments mis en jeu dans le problème (typiquement $N = 2^n$ si n est le nombre d'éléments), ce qui fait que vis à vis d'un ordinateur classique le résolvant en 2^n , l'algorithme de Grover ne permet que de le résoudre en $\sqrt{2^n} = 2^{\frac{n}{2}}$ qui demeure exponentiel.

On peut citer plusieurs problèmes type pouvant être résolus par l'algorithme de Grover, comme les sudokus, les problèmes de satisfiabilité, les problèmes de coloration de graphes, etc.

11.1 Principe

L'algorithme de Grover est constitué de trois étapes principales, comme illustré sur la figure 11.1. Il y a tout d'abord la préparation de l'espace de recherche, qui correspond à toutes les solutions possibles du problème, puis l'application d'un oracle qui permet de marquer les solutions valides, et enfin l'application d'un opérateur de diffusion qui permet d'amplifier l'effet de l'oracle afin de pouvoir mesurer la solution à la fin de l'algorithme.

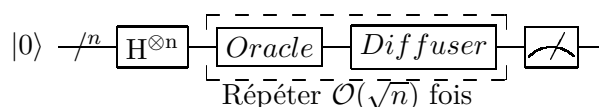


FIGURE 11.1 : Principe de l'algorithme de Grover

Au début, on obtient un espace de recherche qui correspond à une superposition de tous les états possibles, $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$, avec une même probabilité, $\frac{1}{N} = \frac{1}{2^n}$, d'obtenir n'importe quel état. Dans ce cas, on a une chance sur 2^n de mesurer la bonne solution, soit en moyenne 2^{n-1} essais pour trouver la solution.

La procédure d'amplification vise à augmenter l'amplitude de la bonne solution, et diminuer celle des mauvaises solutions, jusqu'à avoir une probabilité presque certaine. De plus, cet algorithme a une jolie interprétation géométrique, via des réflexions qui entraînent des rotations dans le plan.

Considérons donc deux vecteurs $|w\rangle$ et $|s\rangle$, avec $|w\rangle$ le vecteur correspondant aux solutions valides, et $|s\rangle$ le vecteur de départ. Ils forment donc un plan à deux dimensions dans l'espace de Hilbert \mathbb{C}^N . Ils ne sont néanmoins pas orthogonaux, et on peut donc définir un vecteur $|s'\rangle$ en enlevant $|w\rangle$ à $|s\rangle$ et en normalisant le résultat.

1. On commence l'algorithme par une superposition uniforme de tous les états $|s\rangle$.

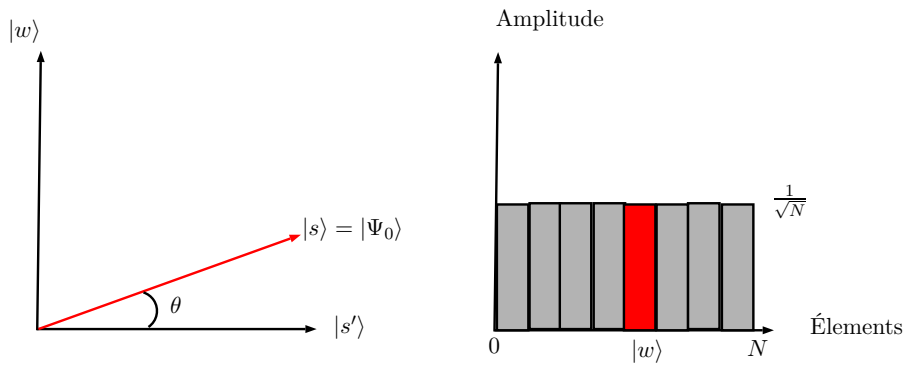


FIGURE 11.2 : État après initialisation

Le graphique de gauche correspond au plan défini par $|w\rangle$ et $|s'\rangle$ qui permet d'exprimer l'état initial $|s\rangle = \sin \theta |w\rangle + \cos \theta |s'\rangle$, avec $\theta = \arcsin \langle s|w\rangle = \arcsin \frac{1}{\sqrt{N}}$. Le graphique de droite montre les amplitudes de l'état $|s\rangle$.

2. On applique ensuite l'oracle U_f .

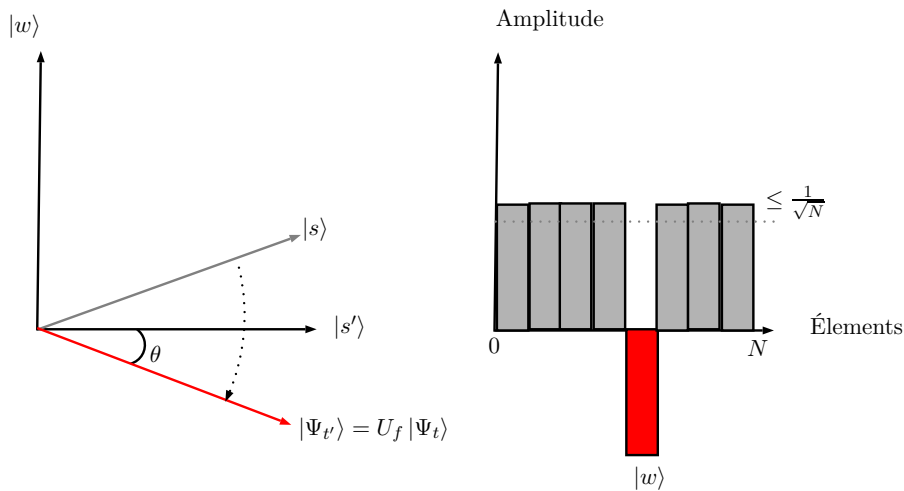


FIGURE 11.3 : État après l'oracle

Géométriquement, cela correspond à une réflexion de $|s\rangle$ autour de $|s'\rangle$. De fait, l'amplitude de $|w\rangle$ devient négative, l'amplitude moyenne diminue donc.

3. On applique finalement le diffuseur U_s , qui correspond à une réflexion autour de $|s\rangle$.

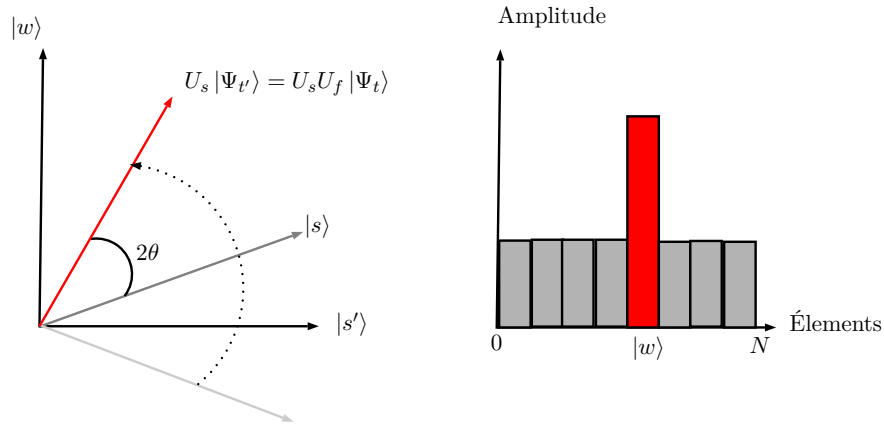


FIGURE 11.4 : État après le diffuseur

Les deux réflexions successives correspondent toujours à une rotation dans le plan. La transformation $U_s U_f$ va donc faire tourner le vecteur $|s\rangle$ plus près de $|w\rangle$. L'action de U_s peut être prise comme une réflexion sur l'amplitude moyenne, ce qui va booster l'amplitude de $|w\rangle$ qui est négative et lui faire prendre près de trois fois sa valeur précédente, tout en diminuant les autres amplitudes.

On répète ensuite les deux dernières étapes à plusieurs reprises, jusqu'à ce que la probabilité de mesurer $|w\rangle$ soit suffisamment grande.

Après t étapes, nous sommes donc dans l'état $|\Psi_t\rangle = (U_s U_f)^t |s\rangle$. Pour trouver le nombre idéal t d'étapes, on peut utiliser la formule suivante :

$$t = \left\lfloor \frac{\pi}{4} \sqrt{\frac{N}{m}} \right\rfloor$$

où m est le nombre de solutions valides, et N la taille de l'espace de recherche. Cette formule est démontrable via la représentation géométrique de l'algorithme.

Pour l'oracle U_f , il faut créer un opérateur qui inverse le signe si l'état sur lequel on l'applique est une solution valide. On peut le faire à partir d'une fonction qui vaut 0 si l'état n'est pas une solution valide, et 1 sinon, et via le retour de phase, comme pour l'algorithme de Deutsch-Jozsa, on peut la transformer en $U_f |x\rangle = (-1)^{f(x)} |x\rangle$.

Le diffuseur U_s peut être construit très simplement, en appliquant d'abord les mêmes portes que pour initialiser l'état $|s\rangle$, ce qui a pour effet de le ramener à l'état $|0\rangle$, et il suffit donc de créer une symétrie par rapport à $|0\rangle$, ce qui peut être fait avec en encadrant une porte multicontrolée Z avec des portes X . Finalement, on remet le circuit dans l'état du début en appliquant l'inverse de l'initialisation.

11.2 Comparaison avec une implémentation classique

Nous pouvons appliquer l'algorithme de Grover avec différentes tailles d'entrées, et comparer avec une implémentation classique. L'implémentation classique consiste à tester toutes les solutions possibles, et à s'arrêter quand on en trouve une qui fonctionne. Celle quantique est similaire à celle classique car il faut une logique similaire pour l'oracle, mais il intègre ensuite le diffuseur. Afin de faciliter la compréhension, nous choisirons des oracles faciles à implémenter.

11.2.1 2 entrées

Avec deux entrées, nous allons créer un circuit qui détecte si les deux entrées sont à 1. L'oracle sera donc simplement une CZ qui change le signe de l'état si les deux entrées sont à 1. Le circuit d'amplification sera également comme décrit précédemment, avec une initialisation à l'aide de portes d'Hadamard.

Afin de connaître le nombre d'itérations à effectuer, nous allons utiliser la formule de l'angle $\theta = \arcsin \frac{1}{\sqrt{N}}$, avec $N = 4$. Ainsi $\theta = \frac{\pi}{6}$, et après t étapes, nous avons l'état $(U_s U_f)^t = \sin \theta_t |w\rangle + \cos \theta_t |s'\rangle$ et l'on sait que $\theta_t = (2t+1)\theta$ par la figure 11.4. Pour obtenir une probabilité de mesure de $|w\rangle$ de 1, il faut donc $\theta_t = \frac{\pi}{2}$, ce qui donne connaissant $\theta = \frac{\pi}{6}$ que $t = 1$. On voit avec cela que l'on aura une réponse exacte en une seule itération, qui est la mesure de $|11\rangle$.

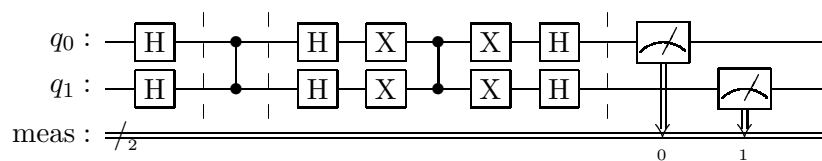


FIGURE 11.5 : Algorithme de Grover avec 2 entrées, sortie 11

On peut également préciser que la recherche classique se ferait avec une porte AND entre les deux entrées, et que l'on doit les tester toutes si on veut être sûr de trouver la bonne, et en moyenne la moitié avec de la chance.

11.2.2 3 entrées

Dans le circuit suivant, on remarque les mêmes patterns que dans le circuit précédent.

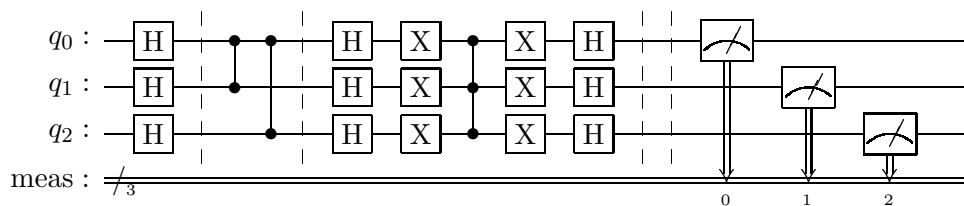


FIGURE 11.6 : Algorithme de Grover avec 3 entrées

Pour le parallèle avec l'ordinateur classique, on va essayer de trouver les sorties escomptées. On voit que ce coup-ci, l'oracle est composé de deux CZ , l'une entre les deux premières entrées, et l'autre entre la première et la troisième. De fait, si seuls les deux premiers qubits sont à 1, alors le premier CZ va inverser la phase et cela devient un résultat possible. De même pour la seconde porte, et finalement si les deux agissent en même temps, alors les effets s'annulent et on se retrouve avec une phase inchangée, ce qui fait que ce n'est pas une des solutions possibles. De fait, on en conclut que l'on devrait mesurer soit $|011\rangle$, soit $|101\rangle$. De plus, on peut calculer

$t = \lfloor \frac{\pi}{4} \sqrt{\frac{N}{m}} \rfloor = 1$ avec $N = 2^3$ et $m = 2$. La simulation du circuit nous donne bien les deux résultats désirés.

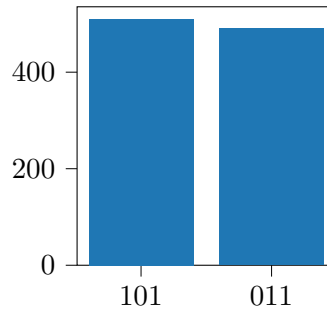


FIGURE 11.7 : Mesure de l'algorithme de Grover avec 3 entrées

11.2.3 4 entrées

Pour celui à 4 entrées, ça se complique. Considérons le circuit suivant, qui a un oracle dont on ne connaît pas le nombre de solutions.

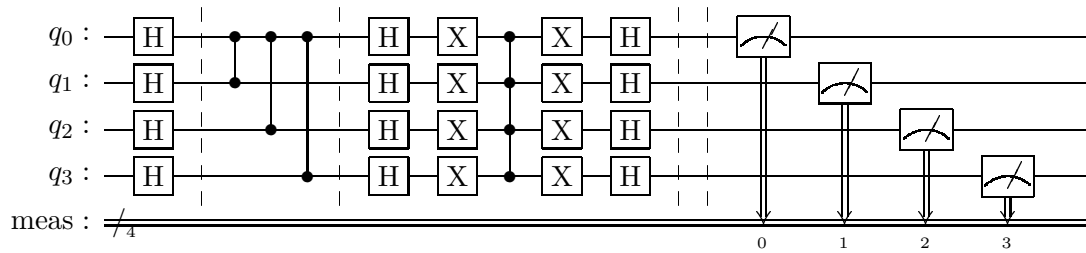


FIGURE 11.8 : Algorithme de Grover avec 4 entrées

Rappelons-nous que l'application $U_s U_f$ fait une rotation d'un angle de 2θ . Or, on sait retrouver l'angle d'un opérateur en utilisant l'estimation de phase. De fait, en créant un circuit avec cette opération que l'on contrôle, on peut connaître l'angle de rotation provoqué par l'opérateur [68].

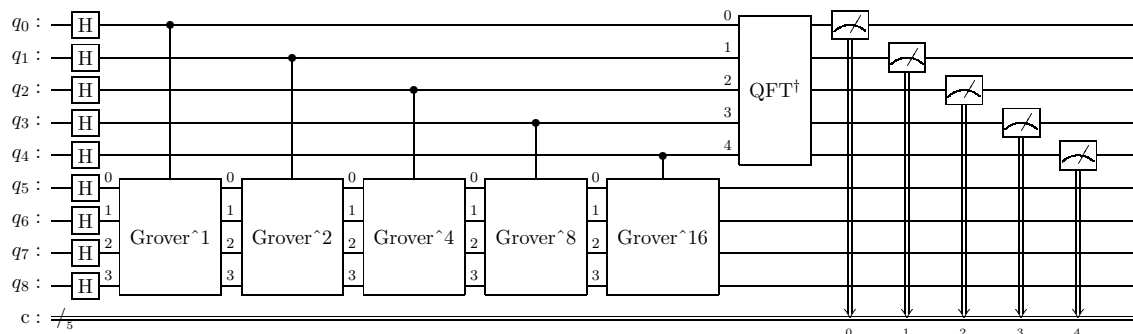


FIGURE 11.9 : Recherche de l'angle de rotation de l'opérateur $U_s U_f$

En voyant les résultats de la simulation, on peut voir que l'angle ne semble pas être une valeur que l'on peut calculer exactement avec peu de ressources.

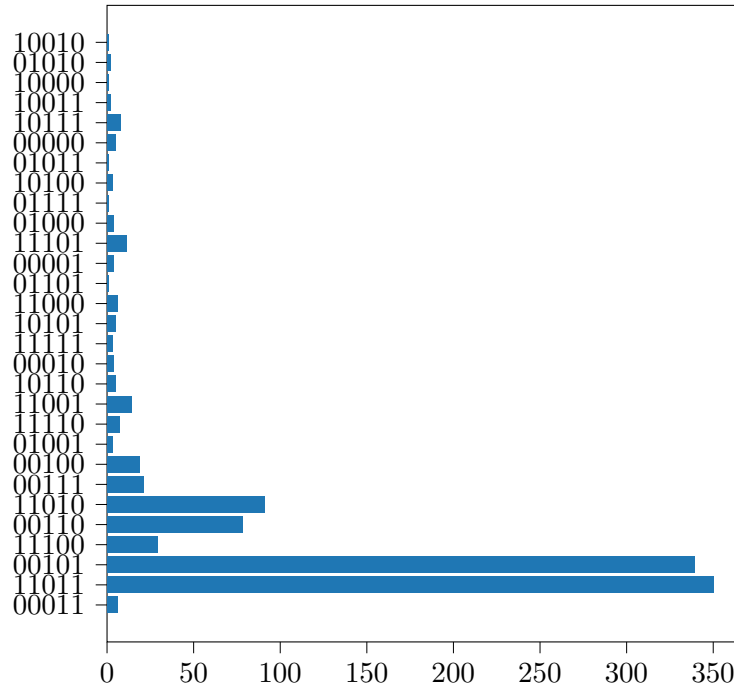


FIGURE 11.10 : Simulation de l'estimation de phase de l'opérateur de Grover

Néanmoins ces résultats sont suffisants pour pouvoir calculer le bon nombre de solutions possibles. On observe donc que le nombre le plus mesuré est 27 (11011 en base 2). La fraction de l'angle est donc $\frac{27}{25}$, que l'on multiplie par 2π pour obtenir l'angle de rotation de l'opérateur $U_s U_f$, i.e. $2\theta = \frac{27}{25}2\pi \approx 5.3$. Finalement, pour obtenir le nombre de solutions m , on reprend la figure 11.2. On constate alors que par définition, $|s\rangle = \sqrt{\frac{m}{N}}|w\rangle + \sqrt{1 - \frac{m}{N}}|s'\rangle$, de plus $|s\rangle = \sin\theta|w\rangle + \cos\theta|s'\rangle$. Ainsi nous pouvons conclure que $\langle s'|s\rangle = \sqrt{1 - \frac{m}{N}} = \cos\theta \Leftrightarrow m = N \sin^2\theta$. De la valeur de 2θ calculée précédemment, on peut donc déduire que $m \approx 3.6$ d'où l'on peut penser que le nombre de solutions est 4.

De cette conclusion, le calcul du nombre d'itérations de l'algorithme de Grover est donc $t = \lfloor \frac{\pi}{4} \sqrt{\frac{2^4}{4}} \rfloor = 1$. En faisant une simulation du circuit présenté en figure 11.8, on obtient bien 4 solutions différentes, qui sont bien les 4 solutions de l'oracle utilisé.

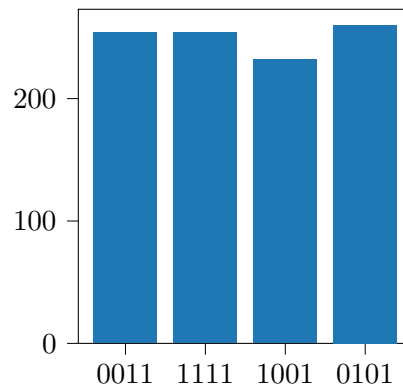


FIGURE 11.11 : Résultats du circuit 11.8

11.2.4 Différence de complexité

On voit donc que l'algorithme de Grover permet de trouver une solution à un problème de manière plus efficace que de manière classique. Il y a tout d'abord le nombre d'opérations à effectuer qui est réduit, de manière classique en moyenne la moitié des possibilités doivent être testées donc 2^{n-1} opérations, alors que cette méthode permet de le faire en $2^{\frac{n}{2}}$, ce qui est non-négligeable quand n est grand. On peut également noter d'autres avantages. Par exemple, connaître le nombre de solutions n'est pas possible de manière classique à moins de tester toutes les solutions possibles, alors que via ce que l'on a vu, on peut le faire avec un protocole standard avec un ordinateur quantique.

11.3 Résolution d'un sudoku

Un autre exemple, assez récréatif, est la résolution d'un sudoku. En effet, un sudoku est un problème de satisfiabilité, et donc résoluble par l'algorithme de Grover. Prenons la grille suivante, qui est un sudoku trivial pour illustrer le principe.

1	<i>2</i>	<i>3</i>
2	<i>3</i>	1
<i>3</i>	1	<i>2</i>

FIGURE 11.12 : Sudoku à résoudre (réponses attendues en italique)

Notons que la grille est choisie de sorte à éviter d'encoder chaque nombre sur 2 bits, car tous les 1 sont placés, ce qui serait au-delà des ressources disponibles, car cela demanderait soit un superordinateur, soit un ordinateur quantique à la pointe de la technologie.

On va donc établir une convention pour encoder les nombres : les 2 vont apparaître comme des 0 et les 3 comme des 1 dans la sortie. De plus, la sortie va donner les nombres par ligne de haut en bas et de gauche à droite, soit le bit de gauche de la sortie correspond au nombre en haut à gauche de la grille.

L'initialisation du circuit se fait en appliquant des portes d'Hadamard sur tous les qubits qui encodent les nombres à trouver. Notons également le dernier qubit qui est initialisé à $|-\rangle$ afin de créer un retour de phase permettant de marquer les solutions valides et donc de créer l'oracle.

L'oracle est constitué de porte CX qui vérifient les différentes contraintes du sudoku. On va donc avoir sur la première vérification de condition une CX avec le qubit correspondant à en bas à droite de la grille, et une avec le qubit correspondant au nombre en bas à gauche. De fait, si les deux sont différents comme attendu, le qubit de vérification sera inversé, et sinon il demeurera à 0. Nous faisons de même pour les autres contraintes, et nous obtenons donc un qubit de vérification par contrainte. Pour les cases qui dépendent de données déjà présentes dans la grille, nous initialisons le qubit de vérification à la valeur de la case, et nous appliquons une CX avec le qubit qui en dépend.

Pour finir, nous utilisons une porte multicontrôlée X pour inverser le qubit visant à créer le retour de phase si tous les qubits de vérification sont à 1, ce qui correspond à une solution valide, puis nous réappliquons toutes les portes CX précédentes pour remettre les qubits de vérification à 0.

Le diffuseur est construit de manière similaire aux précédents, outre l'implémentation de

la porte multicontrôlée Z qui est faite via une X dont la cible est encadrée par des portes H .

Finalement, de par l'unicité des solutions du sudoku, nous savons que nous n'aurons qu'une seule solution, et donc il faut appliquer l'algorithme de Grover $t = \lfloor \frac{\pi}{4} \sqrt{2^5} \rfloor = 4$. Ce qui nous donne après simulation des résultats presque parfaits.

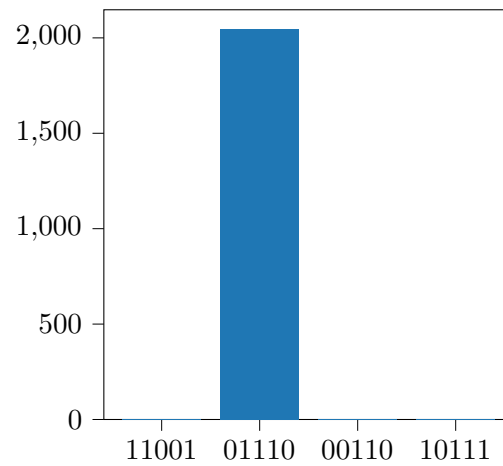


FIGURE 11.14 : Résultat de l'algorithme de Grover pour le sudoku 11.12

En effet, selon les conventions que nous avons établies, la liste 01110 correspond à la solution attendue 23332.

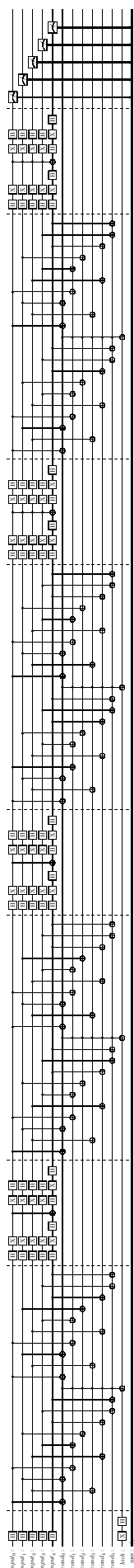


FIGURE 11.13 : Circuit de résolution du sudoku 11.12

Chapitre 12

Estimation de phase itérative

Durant les chapitres précédents, l'estimation de phase a eu de nombreux rôles très utiles, que ce soit dans l'implémentation de l'algorithme de Shor ou dans la recherche du nombre de solutions de l'opérateur de Grover. Néanmoins, ce circuit grossit assez vite quand on désire une précision élevée, ce qui entraîne une augmentation de l'erreur dans ces grands circuits.

Nous allons donc voir dans ce chapitre une méthode permettant d'obtenir une estimation de phase avec une précision arbitraire, tout en gardant un circuit de taille constante. L'estimation de phase itérative [69] est une méthode consistante à faire certaines opérations, puis à mesurer le circuit, et à recommencer en suivant une certaine procédure jusqu'à obtenir la précision désirée.

Reprenons donc le problème de l'estimation de phase 8.3, et considérons que l'angle θ peut être écrit comme une $\theta = \frac{\theta_1}{2} + \frac{\theta_2}{4} + \dots + \frac{\theta_m}{2^m}$. Nous allons développer la méthode avec une porte U active sur un qubit, mais il est aussi valable pour une porte U active sur plusieurs qubits.

L'initialisation du circuit se fait avec deux qubits, $q_0 \rightarrow |+\rangle$ et $q_1 \rightarrow |\Psi\rangle$, avec $|\Psi\rangle$ un vecteur propre de U . L'application d'une porte CU un nombre 2^t de fois va, par un retour de phase, faire passer $q_0 \rightarrow |0\rangle + e^{2i\pi\theta 2^t} |1\rangle$.

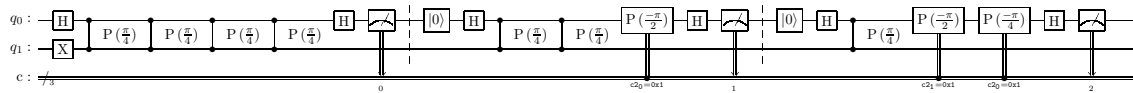
Alors, pour $t = m - 1$, nous obtenons un facteur pour $|1\rangle$ de $e^{2i\pi\theta 2^{m-1}} = e^{2i\pi 2^{m-1}(\frac{\theta_1}{2} + \frac{\theta_2}{4} + \dots + \frac{\theta_m}{2^m})} = e^{2i\pi \frac{\theta_m}{2}}$, car les autres facteurs sont des multiples de 2π . De fait, si $\theta_m = 0$, alors le facteur est égal à 1, et si $\theta_m = 1$, alors le facteur est égal à -1 , ce qui fera basculer le qubit q_0 à $|+\rangle$ ou $|-\rangle$. En appliquant alors une porte de Hadamard sur q_0 , nous obtenons $|0\rangle$ ou $|1\rangle$, qui correspond à θ_m .

Pour la deuxième itération, nous allons appliquer CU un nombre 2^{m-2} de fois, ce qui va nous donner un facteur pour $|1\rangle$ de $e^{2i\pi\theta 2^{m-2}} = e^{2i\pi 2^{m-2}(\frac{\theta_1}{2} + \frac{\theta_2}{4} + \dots + \frac{\theta_m}{2^m})} = e^{2i\pi \frac{\theta_{m-1}}{2}} e^{2i\pi \frac{\theta_m}{4}}$. Comme nous connaissons déjà θ_m , nous pouvons annuler le facteur le concernant en appliquant une porte de phase de $-\frac{\pi}{2}$ sur q_0 si $\theta_m = 1$, et sinon ne rien faire.

On continue la même procédure jusqu'à la dernière itération, en enlevant à chaque fois les facteurs déjà connus.

Il suffit ensuite pour connaître la phase de prendre la liste des bits $\theta_1\theta_2\dots\theta_m$ et de les convertir en décimal, puis de diviser par 2^m pour obtenir la phase θ .

Par exemple, pour la porte $U = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}$, nous pouvons nous attendre à obtenir $\theta = \frac{1}{8} = 0.125$.

FIGURE 12.1 : Recherche par itération de la phase de U

Les résultats obtenus montrent bien le résultat attendu, et il est à noter la faible proportion d'erreurs sur un ordinateur quantique réel. Notons que dans ce cas, nous avons été contraints d'utiliser un modèle de bruit plutôt qu'un vrai ordinateur quantique, car le contrôle d'une porte par la mesure d'un qubit n'est pas encore implémenté sur les ordinateurs quantiques à notre disposition.

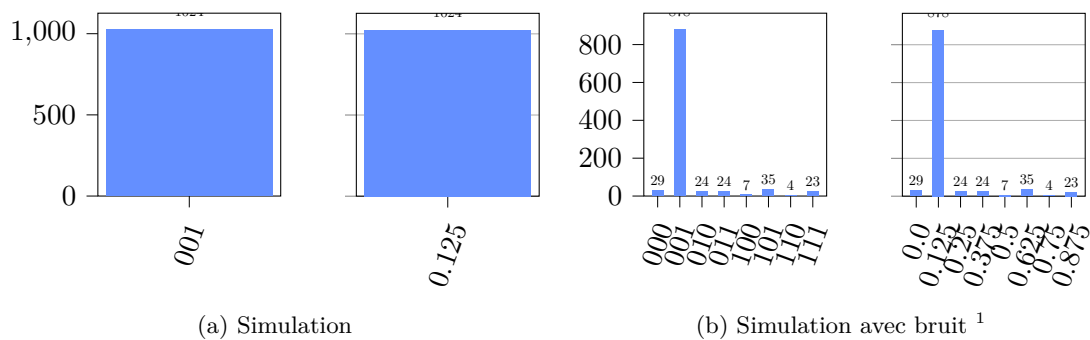


FIGURE 12.2 : Résultats du circuit (chaîne de bits | valeur décimale)

Cette manière de faire permet donc d'obtenir une estimation de phase avec une précision arbitraire sans nécessiter un plus gros circuit. Cela rend donc son utilisation possible sur un ordinateur actuel ou à court terme. Dans cet objectif, il permet aussi de limiter le temps d'intrication des qubits, car ils sont mesurés à chaque itération, ce qui permet d'utiliser cet algorithme sans nécessiter un temps de cohérence trop long.

¹Exécuté selon le modèle *FakeMelbourne*, à date du 20.08.2023, calqué sur un ordinateur d'IBM hors-ligne au moment de la rédaction

Chapitre 13

Modélisation d'un système physique

La modélisation de systèmes quantiques fut une des premières applications pour lequel fut pensé l'ordinateur quantique. En effet, la simulation d'un système physique est un problème qui peut être résolu par un ordinateur quantique de manière plus efficace qu'un ordinateur classique, car il utilise des phénomènes similaires à ceux qui sont étudiés.

Dans ce chapitre, nous allons voir comment calculer les niveaux d'énergie hyperfins d'un atome d'hydrogène à l'état fondamental. L'atome d'hydrogène est composé d'un proton et d'un électron, et chacun de ces deux éléments possède un spin, qui prend une valeur positive ou négative. Il y a donc quatre états de spins de l'atome d'hydrogène, soit les deux spins sont positifs (état $|++\rangle$), soit les deux spins sont négatifs (état $|--\rangle$), soit le spin de l'électron est positif et celui du proton est négatif (état $|+-\rangle$), soit le spin de l'électron est négatif et celui du proton est positif (état $| - + \rangle$).

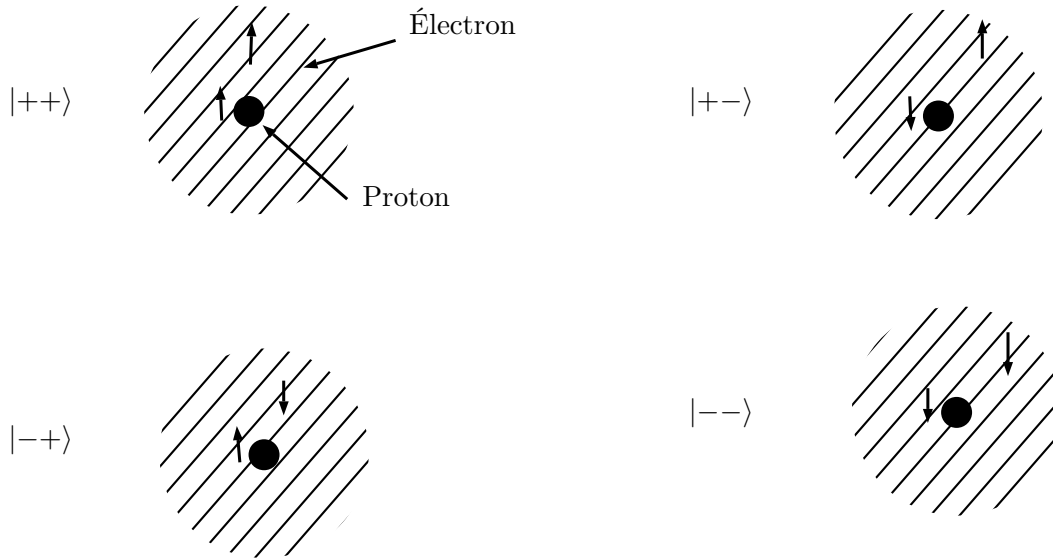


FIGURE 13.1 : États de base de l'état fondamental de l'atome d'hydrogène

De cela, il demeure la question de savoir comment les particules interagissent entre elles et donc comment le système évolue. Afin de le faire, nous allons utiliser l'équation de Schrödinger, qui a été présentée au début afin de justifier les propriétés quantiques de la matière, puis fut quelque peu oubliée. C'est donc à ce moment que nous allons la réintroduire :

$$-\frac{\hbar^2}{2m} \frac{\partial^2 \Psi(x, t)}{\partial x^2} + V(x, t) \Psi(x, t) = i\hbar \frac{\partial \Psi(x, t)}{\partial t}$$

qui, formulée comme cela, est assez peu pratique dans notre cas. En effet, toute la partie gauche de l'équation peut être exprimée comme un opérateur d'évolution, nommé un hamiltonien, et noté \hat{H} . Cela donne donc :

$$\hat{H} |\Psi(t)\rangle = i\hbar \frac{\partial |\Psi(t)\rangle}{\partial t}$$

avec l'hamiltonien qui dépend généralement du temps. Quand ce dernier ne dépend pas du temps, l'équation se simplifie en :

$$\hat{H} |\Psi\rangle = E |\Psi\rangle$$

avec E l'énergie du système.

Petite note sur l'hamiltonien en mécanique classique. Celui-ci est étroitement lié au lagrangien, qui est une manière de décrire un système physique. Considérant une position q , la variation de celle-ci par rapport au temps est donnée par :

$$\dot{q} = \frac{dq}{dt}$$

qui permet de définir la quantité de mouvement p comme étant :

$$p = \frac{\partial \mathcal{L}}{\partial \dot{q}}$$

avec \mathcal{L} une fonction appelée lagrangien. De cela, on peut établir une équation différentielle, appelée équation d'Euler-Lagrange, qui est parfois plus simple à résoudre que l'équation de Newton, qui s'écrit :

$$\frac{dp}{dt} = \frac{\partial \mathcal{L}}{\partial q}$$

L'hamiltonien \mathcal{H} est défini quant à lui comme étant la somme de l'énergie cinétique et de l'énergie potentielle (de manière rigoureuse, il s'agit de la transformée de Legendre du lagrangien, néanmoins nous nous suffirons de la définition précédente). Il sert ensuite à définir les équations d'Hamilton :

$$\dot{q} = \frac{\partial \mathcal{H}}{\partial p} \quad \dot{p} = -\frac{\partial \mathcal{H}}{\partial q}$$

et il a un rôle similaire en mécanique quantique.

Quand on parle d'un opérateur d'évolution qui influe la fonction d'onde, ça rappellera les portes des ordinateurs quantiques. Celles-ci sont en effet des opérateurs d'évolution, et en pratique on calcule l'hamiltonien de chaque qubit afin de le modifier via les micro-ondes afin que l'utilisateur final ne voie que les portes quantiques sans devoir faire ces calculs.

Le système que l'on étudie est un système à deux particules, et chaque particule a deux états, ce qui rappelle les qubits. Or l'hamiltonien du système peut être exprimé selon les bases de Pauli, qui sont les matrices des portes quantiques X , Y et Z . Dans ce cas précis, l'hamiltonien est donné par :

$$\hat{H} = A(XX + YY + ZZ)$$

avec A une constante propre à l'atome d'hydrogène qui peut être calculée via les flux de champs magnétiques, et qui vaut environ $1.47 \cdot 10^{-6}$ électrons-volts (eV). Comme évoqué pour le cas classique, l'hamiltonien est étroitement lié à l'énergie du système, et par l'équation de Schrödinger indépendante du temps, on peut déduire que l'énergie du système correspond aux valeurs propres de l'hamiltonien, qui peut être fait via sa mesure dans les différentes bases :

$$E = \langle H \rangle = A(\langle XX \rangle + \langle YY \rangle + \langle ZZ \rangle)$$

Cette opération est faisable avec un ordinateur quantique en faisant les changements de base nécessaires. Voyons donc ce que cela donne déjà avec une porte :

$$\begin{aligned}\langle Z \rangle &= \langle q|Z|q \rangle = \langle q|0 \rangle \langle 0|q \rangle - \langle q|1 \rangle \langle 1|q \rangle = |\langle 0|q \rangle|^2 - |\langle 1|q \rangle|^2 \\ \langle X \rangle &= \langle q|X|q \rangle = \langle q|+\rangle \langle +|q \rangle + \langle q|-\rangle \langle -|q \rangle = |\langle +|q \rangle|^2 + |\langle -|q \rangle|^2 = H(|\langle 0|q \rangle|^2 - |\langle 1|q \rangle|^2) \\ \langle Y \rangle &= \langle q|Y|q \rangle = -i\langle q|0 \rangle \langle 1|q \rangle + i\langle q|1 \rangle \langle 0|q \rangle = -i\langle X \rangle = S^\dagger H(|\langle 0|q \rangle|^2 - |\langle 1|q \rangle|^2)\end{aligned}$$

où l'on calcule en prenant un qubit général $|q\rangle$ et l'on cherche à isoler les mesures selon $|0\rangle$ et $|1\rangle$ en utilisant des portes. Notons que la porte S^\dagger est la porte inverse de S , qui est définie comme étant :

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

d'où le fait que S^\dagger soit définie comme étant :

$$S^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix}$$

Si l'on fait ensuite de même avec le composante de l'hamiltonien, on obtient :

$$\begin{aligned}\langle ZZ \rangle &= \langle \psi|ZZ|\psi \rangle = \langle \psi|(|0\rangle\langle 0| - |1\rangle\langle 1|) \otimes (|0\rangle\langle 0| - |1\rangle\langle 1|)|\psi \rangle \\ &= |\langle 00|\psi \rangle|^2 - |\langle 01|\psi \rangle|^2 - |\langle 10|\psi \rangle|^2 + |\langle 11|\psi \rangle|^2 \\ \langle XX \rangle &= \langle \psi|XX|\psi \rangle = \langle \psi|H(|0\rangle\langle 0| - |1\rangle\langle 1|) \otimes H(|0\rangle\langle 0| - |1\rangle\langle 1|)|\psi \rangle \\ &= H^{\otimes 2}(|\langle 00|\psi \rangle|^2 - |\langle 01|\psi \rangle|^2 - |\langle 10|\psi \rangle|^2 + |\langle 11|\psi \rangle|^2) \\ \langle YY \rangle &= \langle \psi|YY|\psi \rangle = \langle \psi|S^\dagger H(|0\rangle\langle 0| - |1\rangle\langle 1|) \otimes S^\dagger H(|0\rangle\langle 0| - |1\rangle\langle 1|)|\psi \rangle \\ &= S^{\dagger \otimes 2} H^{\otimes 2}(|\langle 00|\psi \rangle|^2 - |\langle 01|\psi \rangle|^2 - |\langle 10|\psi \rangle|^2 + |\langle 11|\psi \rangle|^2)\end{aligned}$$

pour un état général $|\psi\rangle$.

La méthode ensuite consiste à créer les quatre états de Bell, qui sont les quatre états de superposition équilibrés pour un système à deux qubits. Ensuite pour chacun de ces états, on mesure les composantes selon les différentes bases, puis on calcule l'énergie du système via les proportions de chacune des mesures.

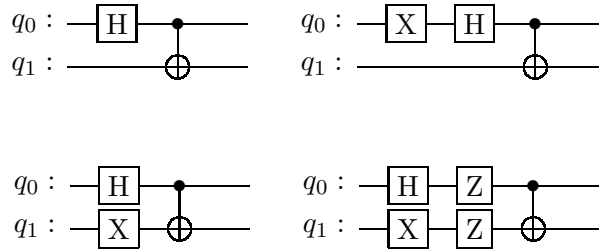
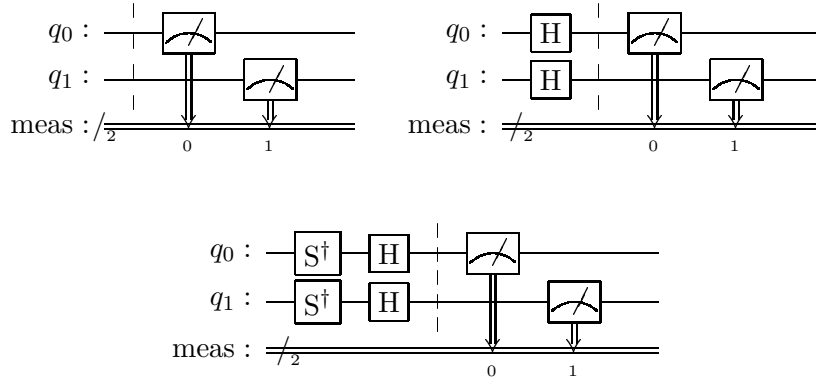


FIGURE 13.2 : Création des états de Bell $|\Phi^+\rangle$, $|\Phi^-\rangle$, $|\Psi^+\rangle$ et $|\Psi^-\rangle$

FIGURE 13.3 : Circuit de mesure selon la base ZZ , XX , YY

Les états de Bell sont traités ensuite en les mesurant selon les bases ZZ , XX et YY , 2^{16} fois pour chaque base. Nous dénomons p_i la proportion de mesure de l'état i , soit $p_i = \frac{\#i}{2^{16}}$ avec $i \in \{00, 01, 10, 11\}$. Ensuite, on sait que l'énergie est donnée par la formule suivante, déduite des équations présentées plus haut :

$$E = A(p_{00} - p_{01} - p_{10} + p_{11})$$

ce qui nous donne par simulation que pour les états $|\Phi^+\rangle$, $|\Phi^-\rangle$ et $|\Psi^+\rangle$, l'énergie est de $E \approx 1.47 \cdot 10^{-6}$ [eV] et pour l'état $|\Psi^-\rangle$, l'énergie est de $E \approx -4.41 \cdot 10^{-6}$ [eV]. Cela correspond bien aux valeurs théoriques attendues, de $E = A$ pour les trois premiers états et $E = -3A$ pour le dernier état [70].

Cette différence d'énergie est visible par exemple dans les radiations émises par les atomes d'hydrogène, qui crée une onde particulière de 21 [cm] de longueur d'onde. Cette onde peut être déduite de la formule $E = \hbar f$ avec E la différence d'énergie entre ces états hyperfins, soit $E = 4A$ et f la fréquence de l'onde émise. Les radiations se propageant à la vitesse de la lumière, on peut déduire la longueur d'onde via $f = \frac{c}{\lambda}$ avec c la vitesse de la lumière et λ la longueur d'onde. Ainsi, on obtient $\omega = \frac{c}{\lambda} = \frac{ch}{E} = \frac{ch}{4A} \approx 21$ [cm].

Cette simulation simple à de plus l'avantage de pouvoir être menée sur un ordinateur quantique actuel, qui via les technologies de réductions d'erreur permet des résultats avec une différence de $\pm 2\%$ ¹. Cela permet de donner de bons espoirs pour des simulations plus complexes, comme par exemple des molécules ou des propriétés quantiques des matériaux, sur des ordinateurs quantiques dans un futur proche, en ne résolvant plus le système mathématiquement ou en effectuant des calculs qui grossissent de manière exponentielle sur un ordinateur classique, mais en le simulant sur un ordinateur quantique.

¹Résultat basé sur une simulation du bruit d'un ordinateur quantique selon le modèle *FakeVigo*

Quatrième partie

Et après...

Chapitre 14

Technologies de hardware

Les ordinateurs quantiques sont des machines qui sont encore en développement. De fait, de nombreuses technologies sont en compétition pour devenir la technologie dominante.

Quelques critères ont été établis pour la technologie idéale :

- Possibilité de l'agrandir physiquement afin d'augmenter le nombre de qubits
- Les qubits doivent être initialisables à l'état que l'on souhaite
- Les portes doivent être plus rapides que le temps de décohérence
- Le groupe de portes doit être universel
- Les qubits doivent être lisibles facilement

La technologie privilégiée actuellement est celle des qubits supraconducteurs, comme celle utilisée par IBM et Google.

Cependant, d'autres technologies sont étudiées, comme les photons, sur lesquels on peut utiliser au choix leur polarisation, leur présence ou leur absence, ou encore leur temps d'arrivée. On peut aussi utiliser le spin des particules, ou encore comme dans le cas des supraconducteurs, la charge, le courant ou l'énergie d'une jonction Josephson.

Chaque technologie présente des avantages et des inconvénients, ainsi que des défis à relever. Par exemple, les qubits supraconducteurs doivent être contrôlés par l'extérieur, donc il faut par exemple appliquer une tension précise à température ambiante, puis refroidir le tout à des températures proches du zéro absolu, sans pour autant détruire les propriétés de la tension appliquée afin que le qubit reste dans l'état voulu. Les qubits supraconducteurs fonctionnant autour de 20 [mK], il faut dans la procédure décrite au-dessus refroidir de 300 [K] à 20 [mK], ce qui est un défi en soi. Il y a donc été développé un système fonctionnant autour de 4 [K] qui permet de faciliter tout le processus [71].

On peut également citer les qubits de spin, qui eux présentent l'avantage d'avoir une meilleure cohérence, mais qui sont plus difficiles à construire. Ils demeurent assez miniaturisables, et sont contrôlables par des champs magnétiques, ce qui est une différence majeure avec les qubits supraconducteurs. Finalement, leurs conditions de fonctionnement sont plus faciles à atteindre, car ils fonctionnent à des températures plus élevées par exemple.

Quelle que soit la technologie choisie, il y aura toujours la recherche de la réduction des erreurs, car même si les qubits actuels sont fidèles à 99.9%, cela n'est pas suffisant pour faire des calculs complexes, parce que les erreurs s'accumulent et peuvent rendre le résultat faux. De fait, plusieurs méthodes de correction d'erreurs sont étudiées, comme la répétition de qubits, ou différents codes correcteurs d'erreurs [72].

La recherche se fait autour de ces technologies, mais aussi autour de la manière de stocker les qubits, puisqu'il n'existe pas encore de moyen de stocker un qubit de manière stable, comme les RAM pour les bits classiques, ou encore les disques durs. C'est probablement la plus grande difficulté à laquelle les chercheurs sont confrontés, du fait de la décohérence des qubits, qui a déjà des effets sur les temps courts durant l'exécution d'un programme.

Néanmoins, une qRAM est fondamentalement moins importante qu'une RAM classique, car les qubits stockent eux-mêmes l'information, et on peut la déplacer sur un autre via la téléportation quantique, ce qui permet d'avancer que le problème réside dans la capacité de maintien du qubit dans un état stable sur une longue durée.

Comme mentionné également dans la distribution de clés, l'envoi de qubits sur de longues distances est également étudié et quelques solutions ont été trouvées, comme l'envoi de photons sur fibre optique. Cette partie est une des plus développées actuellement, et certaines entreprises le proposent déjà en tant que produit commercial [73]. De plus, on est aussi capable de transmettre des photons intriqués par satellite, ce qui a permis de faire une transmission sur plus de 1200 [km] [74].

D'autres domaines de recherches sont aussi utiles à ce domaine, comme la recherche en matériaux supraconducteurs, car plus ceux-ci sont performants, plus les qubits pourront être améliorés.

Chapitre 15

Sur des machines à court terme

Quelques machines à court terme voir même actuelles permettent de faire tourner des circuits ayant des intérêts pratiques.

Par exemple de nombreuses simulations de phénomènes physiques, que ce soit dans le tout petit comme le partenariat entre le CERN et IBM [75] dans l'étude des particules subatomiques via un ordinateur quantique ou dans le tout grand comme la simulation d'un trou de ver réalisée par les ordinateurs de Google [76].

Ces applications vont se multiplier dans les années à venir, et de plus gros ordinateurs quantiques pourront être utilisés pour simuler des molécules pour la pharmaceutique ou la chimie, mais également des matériaux pour l'industrie.

Un des plus gros espoirs mis dans la technologie quantique est l'intelligence artificielle. En effet, les ordinateurs quantiques pourraient permettre d'entraîner des modèles de machine learning de manière plus efficaces et donc de surpasser les problèmes qui vont se poser bientôt avec les ordinateurs classiques, qui deviennent limités de par leur taille pour continuer à faire des intelligences artificielles plus complexes.

Cela est à plus long terme, mais dans un futur proche, on peut faire travailler des ordinateurs quantiques et classiques ensemble pour faire de l'intelligence artificielle [77]. Malgré les problèmes de décohérence, les ordinateurs quantiques peuvent être utilisés à cette fin, le bruit créé pouvant être même exploité pour modifier les poids d'un réseau de neurones.

Finalement, il s'agit également de déterminer la classe de complexité des ordinateurs quantiques [78], afin de déterminer dans quelle mesure ils sont plus puissants et permettre une meilleure compréhension de leurs capacités.

De plus, on peut noter de nombreuses applications des technologies quantiques au-delà du domaine calculatoire. On a évoqué la transmission d'information, mais on peut également citer la génération de nombres aléatoires par exemple. D'autres applications peuvent paraître plus étonnantes, comme une méthode de mesure de la gravité utilisée ensuite pour détecter l'activité volcanique [79].

Chapitre 16

Sur le long terme

Un ordinateur quantique a des intérêts évidents pour la cryptographie, mais il permet également via des algorithmes comme celui de Grover d'accélérer des problèmes d'optimisation avec des contraintes, comme des trajets de livraison ou des problèmes de planification.

Il y a bien sûr toutes les applications citées dans le chapitre précédent, mais poussée à l'extrême, comme une intelligence artificielle entraînée sur un ordinateur quantique uniquement.

Citons également qu'un canal de communication quantique permettrait de communiquer de manière plus sûre, par exemple via la distribution de clés ou le superdense coding qui empêcherait l'interception d'un message complet.

Si l'on arrivait aussi à créer un stockage quantique, on pourrait enregistrer nos données dessus et y accéder plus rapidement via l'algorithme de Grover par exemple.

Finalement, cela peut être utilisé par exemple pour du calcul pur de fonctions mathématiques, en exploitant par exemple l'apparition intrinsèque de nombres complexes dans les portes quantiques.

Notons que le développement des ordinateurs quantiques et l'avenir qui leur est promis dépend beaucoup du spécialiste que l'on interroge. Par exemple, en introduction, nous citons qu'un qubit parfait pourrait être réalisé à partir de quelques milliers de qubits physiques, mais IBM par exemple citait un article pendant le Quantum Industry Day in Switzerland 2023 qui estimait la possibilité d'atteindre un qubit logique à partir de 50 qubits physiques. En discutant avec Edoardo Charbon, professeur à l'EPFL, il a éclairci ce point en disant que cela dépendait de la qualité recherchée pour le qubit logique, et qu'il serait probablement encore sujet à erreurs. De son point de vue, il ne sait pas encore exactement où cela va mener, que ce soit du côté d'une technologie qui serait dominante ou pas, et dans quel domaine cela se fera précisément. On peut mettre cela en perspective avec le point de vue d'Alain Aspect, déjà présenté dans le chapitre 2, qui lors de sa lecture du 2 octobre 2023 à l'EPFL, voyait déjà une utilité purement scientifique à la recherche sur les ordinateurs quantiques, car cela permet de voir jusqu'où la théorie quantique est vraie, et de la tester.

Chapitre 17

Conclusion

Pour conclure, la technologie de l'ordinateur quantique est encore en développement, mais son avenir est prometteur. Ils pourront révolutionner le monde de l'informatique, et permettre une accélération de la recherche dans de nombreux domaines. Tout en se basant sur les principes assez complexes de la mécanique quantique, une fois ceux-ci acceptés, il est possible d'envisager la puissance de calcul qu'ils permettent, en faisant presque du calcul sur plusieurs valeurs en même temps, à condition de savoir extraire l'information que l'on veut de l'état quantique. Les ordinateurs quantiques sont encore en développement, mais on les voit déjà accessibles pour ceux intéressés, avec un grand enjeu d'apprentissage autour de ce sujet [80], que ce soit les algorithmes et autres applications, ou la réalisation autour du hardware. C'est donc un domaine d'avenir, avec de nombreuses promesses qui commencent à se concrétiser, comme la réalisation d'ordinateurs à plusieurs milliers de qubits qui laissent à espérer un avantage majeur dans les décennies à venir.

Appendices

Annexe A

Utilisation de *Qiskit*

Toutes les expériences présentées dans ce manuscrit ont été réalisées à l'aide de *Qiskit*. C'est un module Python développé par IBM permettant de manipuler des circuits quantiques et de les simuler. Il permet également de les envoyer sur des ordinateurs quantiques réels fournis par IBM, ainsi que sur des simulateurs plus performants que ceux fournis par défaut.

Sur le site se trouvent de nombreux tutoriels et exemples d'utilisation, ainsi que la documentation complète du module. De plus, l'accès aux ordinateurs quantiques est gratuit, il suffit de créer un compte sur le site, qui offre également des serveurs Jupyter pour faire tourner les notebooks, et des outils pour visualiser les circuits quantiques.

Dans cette annexe, nous allons présenter le script utilisé pour construire les circuits quantiques de certains chapitres, et les fonctions utilisées pour les simuler et les envoyer sur des ordinateurs quantiques réels.

Pour cela, nous allons faire la démonstration avec les scripts utilisés pour l'algorithme de Shor, qui ont l'avantage de couvrir de nombreux aspects du module. De plus, la structuration étant en grande partie tirée de la documentation de *Qiskit*, il est d'autant plus facile de s'y retrouver.

Dans un premier temps, nous allons importer les modules nécessaires à la construction du circuit quantique, ainsi qu'à sa visualisation et à sa simulation. De plus, les modules standards du traitement de données sont importés, ainsi que les fonctions mathématiques afin de pouvoir traiter les résultats et faire des calculs dans le script.

Dans cette étape d'initialisation, nous allons également définir la manière dont nous allons simuler le circuit quantique de manière générale sur le script, dans ce cas le simulateur *qasm_simulator* qui est une méthode de simulation tournant localement sur l'ordinateur.

```
In [1]: 1 import matplotlib.pyplot as plt
2 import numpy as np
3 from qiskit import *
4 from qiskit.visualization import plot_histogram
5 from math import gcd
6 from numpy.random import randint
7 import pandas as pd
8 from fractions import Fraction
9
10 from qiskit.visualization import plot_histogram
11 from qiskit.circuit.library import QFT
12
13 import math
14
15 sim = Aer.get_backend('qasm_simulator')
```

Dans les cellules suivantes sont définies les fonctions qui vont être utilisées pour construire le circuit, comme l'exponentiation modulaire, la transformée de Fourier quantique inverse, ou encore les valeurs de a à tester et de qubits de comptage.

```
In [2]: 1 def c_amod15(a, power):
2         """Controlled multiplication by a mod 15"""
3         if a not in [2,4,7,8,11,13]:
4             raise ValueError("'a' must be 2,4,7,8,11 or 13")
5         U = QuantumCircuit(4)
6         for _iteration in range(power):
7             if a in [2,13]:
8                 U.swap(2,3)
9                 U.swap(1,2)
10                U.swap(0,1)
11            if a in [7,8]:
12                U.swap(0,1)
13                U.swap(1,2)
14                U.swap(2,3)
15            if a in [4, 11]:
16                U.swap(1,3)
17                U.swap(0,2)
18            if a in [7,11,13]:
19                for q in range(4):
20                    U.x(q)
21        U = U.to_gate()
22        U.name = f"{a}^{power} mod 15"
23        c_U = U.control()
24        return c_U
```

```
In [3]: 1 N_COUNT = 8 # number of counting qubits
2         a = 7
3         # Specify variables
```

```
In [4]: 1 def qft_dagger(n):
2         """n-qubit QFTdagger the first n qubits in circ"""
3         qc = QuantumCircuit(n)
4         # Don't forget the Swaps!
5         for qubit in range(n//2):
6             qc.swap(qubit, n-qubit-1)
7         for j in range(n):
8             for m in range(j):
9                 qc.cp(-np.pi/float(2**((j-m))), m, j)
10            qc.h(j)
11        qc.name = "QFT+"
12        return qc
```

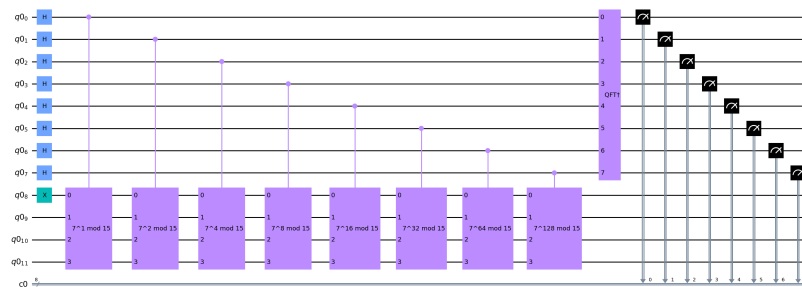
Dans la cellule suivante, nous allons construire le circuit quantique. Nous commençons par créer les registres quantiques et classiques, puis nous initialisons le circuit quantique. Ensuite, nous appliquons les portes désirées pour le circuit, et finalement, nous le dessinons afin de pouvoir le visualiser.

```

In [5]: 1 qr = QuantumRegister(N_COUNT + 4)
2 cr = ClassicalRegister(N_COUNT)
3 qc = QuantumCircuit(qr, cr)
4 # Create QuantumCircuit with N_COUNT counting qubits
5 # plus 4 qubits for U to act on
6
7 # Initialize counting qubits
8 # in state |+>
9 for q in range(N_COUNT):
10     qc.h(q)
11
12 # And auxiliary register in state |1>
13 qc.x(N_COUNT)
14
15 # Do controlled-U operations
16 for q in range(N_COUNT):
17     qc.append(c_amod15(a, 2**q),
18               [q] + [i+N_COUNT for i in range(4)])
19
20 # Do inverse-QFT
21 qc.append(qft_dagger(N_COUNT), range(N_COUNT))
22
23 # Measure circuit
24 qc.measure(range(N_COUNT), range(N_COUNT))
25 qc.draw(fold=-1, output='mpl') # -1 means 'do not fold'

```

Out[5]: 1



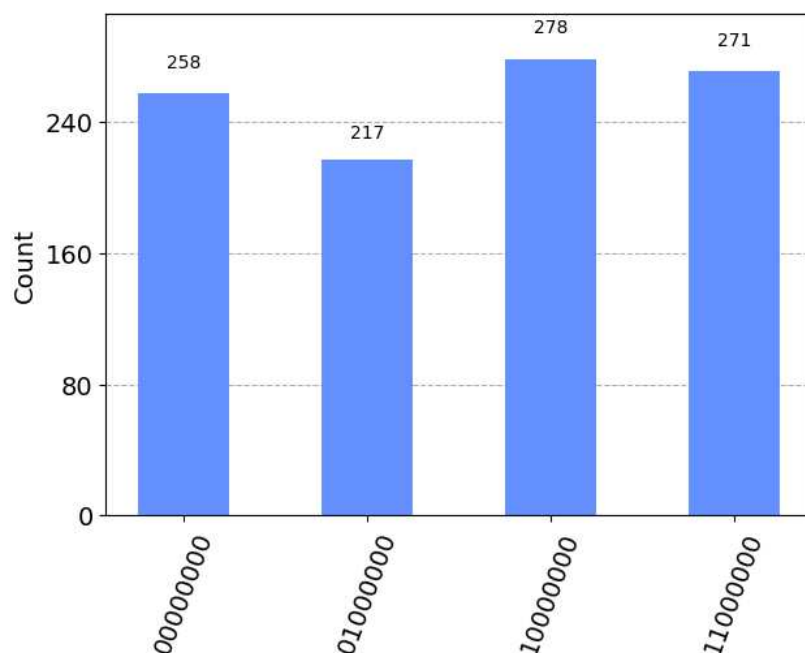
Afin d'étudier son fonctionnement théorique, nous allons simuler le circuit quantique. Pour cela, on utilise le simulateur que l'on définit, puis le circuit est transpilé, donc modifié afin de correspondre aux portes quantiques disponibles sur le simulateur. Le circuit est ensuite exécuté sur le simulateur, et les résultats sont récupérés pour être affichés sous forme d'histogramme.

```

In [6]: 1 aer_sim = Aer.get_backend('aer_simulator')
2 t_qc = transpile(qc, aer_sim)
3 counts = aer_sim.run(t_qc).result().get_counts()
4 plot_histogram(counts)

```


Out[6]: 1



Dans ce cas, les résultats doivent encore être traités afin de pouvoir être exploités. Les cellules suivantes permettent d'effectuer toutes les opérations nécessaires pour pouvoir les utiliser.

```
In [7]: 1 rows, measured_phases = [], []
2 for output in counts:
3     decimal = int(output, 2) # Convert (base 2) string to decimal
4     phase = decimal/(2**N_COUNT) # Find corresponding eigenvalue
5     measured_phases.append(phase)
6     # Add these values to the rows in our table:
7     rows.append([f"{output}(bin) = {decimal:>3}(dec)",
8                 f"{decimal}/{2**N_COUNT} = {phase:.2f}"])
9 # Print the rows in a table
10 headers=["Register Output", "Phase"]
11 df = pd.DataFrame(rows, columns=headers)
12 print(df)
```

	Register Output	Phase
0	10000000(bin) = 128(dec)	128/256 = 0.50
1	00000000(bin) = 0(dec)	0/256 = 0.00
2	11000000(bin) = 192(dec)	192/256 = 0.75
3	01000000(bin) = 64(dec)	64/256 = 0.25

```
In [8]: 1 rows = []
2 for phase in measured_phases:
3     frac = Fraction(phase).limit_denominator(15)
4     rows.append([phase,
5                 f"{frac.numerator}/{frac.denominator}",
6                 frac.denominator])
7 # Print as a table
8 headers=["Phase", "Fraction", "Guess for r"]
9 df = pd.DataFrame(rows, columns=headers)
10 print(df)
```

	Phase	Fraction	Guess	for r
1	0	0.50	1/2	2
2	1	0.00	0/1	1
3	2	0.75	3/4	4
4	3	0.25	1/4	4

Afin d'avoir une idée à quoi cela ressemblerait un tel circuit sur un ordinateur quantique réel, nous pouvons le simuler en utilisant des bruits qui sont calqués sur ceux des ordinateurs quantiques réels.

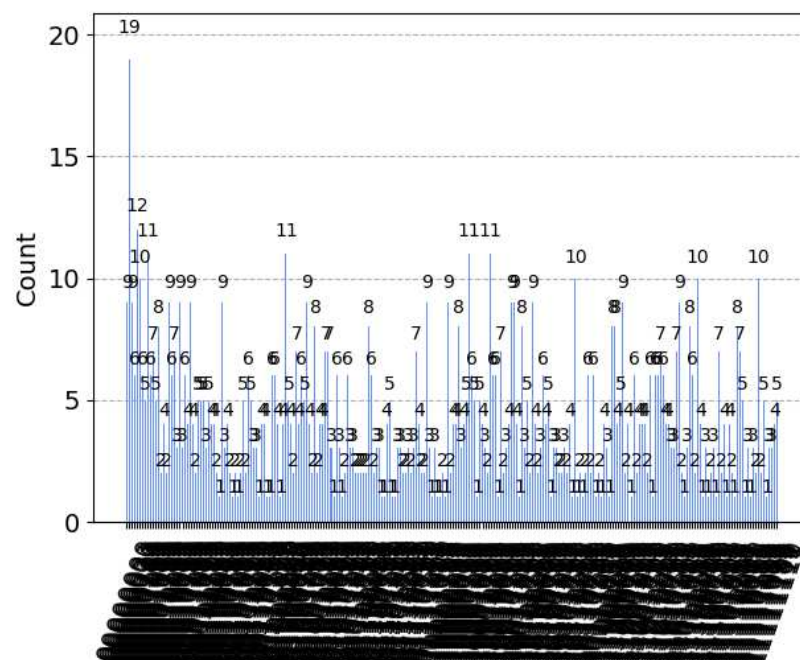
```
In [9]: 1 from qiskit import QuantumCircuit, QuantumRegister, Aer, execute
2 from qiskit.providers.fake_provider import FakeMelbourne
3 backend = FakeMelbourne()
4 shots=1024
```

Dans cette étape, des opérations d'affichage des étapes du programme sont effectuées afin de pouvoir suivre l'avancement de celui-ci, car cela peut prendre un certain temps. Notons que dans ce cas, on voit que le circuit est inexploitable, parce que les résultats sont trop bruités et le graphe est illisible.

```
In [10]: 1 print('Start...')
2 qc_trans = transpile(qc, backend, optimization_level=1)
3 print('Done... (1/?)' )
4 results = execute(qc_trans, backend, shots=shots).result()
5 print('Done... (2/?)' )
6 counts_re = results.get_counts()
7 plot_histogram(counts_re)
```

```
1 Start...
2 Done... (1/? )
3 Done... (2/?)
```

Out[10]:¹



On peut ensuite essayer de corriger les erreurs en utilisant la méthode de correction de mesure

proposée par *Qiskit*. On voit que cela fait ressortir certains résultats, mais que le graphe reste assez bruité.

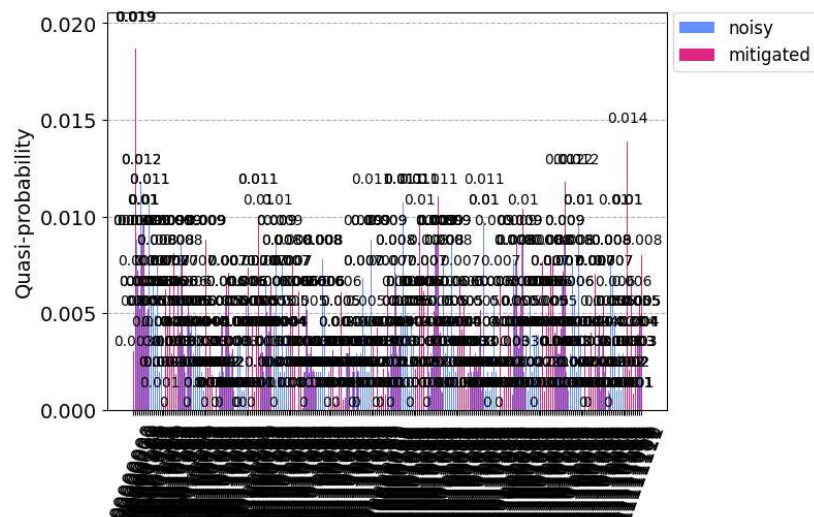
```
In [11]: 1 from qiskit.utils.mitigation import complete_meas_cal,
          2 CompleteMeasFitter
          3 meas_calibs, state_labels = complete_meas_cal(qubit_list
          4 = [0,1,2,3,4,5,6,7], circlabel='mcal')
```

```
In [12]: 1 job = execute(meas_calibs, backend=backend, shots=shots)
          2 cal_results = job.result()
          3
          4 meas_fitter = CompleteMeasFitter(cal_results, state_labels,
          5                               circlabel='mcal')
```

```
In [13]: 1 meas_filter = meas_fitter.filter
          2
          3 mitigated_results = meas_filter.apply(results)
          4 mitigated_counts = mitigated_results.get_counts(0)
```

```
In [14]: 1 plot_histogram([counts_re, mitigated_counts], legend=['noisy', '
          2 mitigated'])
```

```
Out[14]: 1
```



Finalement, le clou du spectacle, nous allons envoyer le circuit sur un ordinateur quantique réel. Dans ce cas, nous utilisons le service *IBMQ*, qui permet d'envoyer des circuits sur les ordinateurs quantiques d'IBM. Afin de garder le token secret, la fonction d'utilisation d'un compte enregistré est utilisée.

```
In [15]: 1 from qiskit import IBMQ
          2 from qiskit.providers.ibmq import least_busy
          3 from qiskit.tools import job_monitor
          4 from qiskit_ibm_runtime import QiskitRuntimeService
          5
          6 shots = 8192
          7 provider = IBMQ.load_account()
          8
          9 service = QiskitRuntimeService()
```

```

1 C:\Users\romai\AppData\Local\Temp\ipykernel_28196\2450165869.py:7:
  DeprecationWarning: The qiskit.IBMQ entrypoint and the qiskit-
  ibmq-provider package (accessible from 'qiskit.providers.ibmq')
  are deprecated and will be removed in a future release. Instead
  you should use the qiskit-ibm-provider package which is
  accessible from 'qiskit_ibm_provider'. You can install it with '
  pip install qiskit_ibm_provider'. Just replace 'qiskit.IBMQ'
  with 'qiskit_ibm_provider.IBMProvider'
2 provider = IBMQ.load_account()

```

Nous créons ensuite le circuit quantique, qui possède 7 qubits, le maximum disponible sur les ordinateurs quantiques mis à disposition gratuitement par IBM.

```

In [16]: 1 N_COUNT = 3 # number of counting qubits
2 a = 7
3 # Specify variables
4
5 # Create QuantumCircuit with N_COUNT counting qubits
6 # plus 4 qubits for U to act on
7 qr = QuantumRegister(N_COUNT + 4)
8 cr = ClassicalRegister(N_COUNT)
9 qc = QuantumCircuit(qr, cr)
10
11 # Initialize counting qubits
12 # in state |+>
13 for q in range(N_COUNT):
14     qc.h(q)
15
16 # And auxiliary register in state |1>
17 qc.x(N_COUNT)
18
19 # Do controlled-U operations
20 for q in range(N_COUNT):
21     qc.append(c_amod15(a, 2**q),
22              [q] + [i+N_COUNT for i in range(4)])
23
24 # Do inverse-QFT
25 qc.append(qft_dagger(N_COUNT), range(N_COUNT))
26
27 # Measure circuit
28 qc.measure(range(N_COUNT), range(N_COUNT))
29
30 # qc.draw(output='mpl')

```

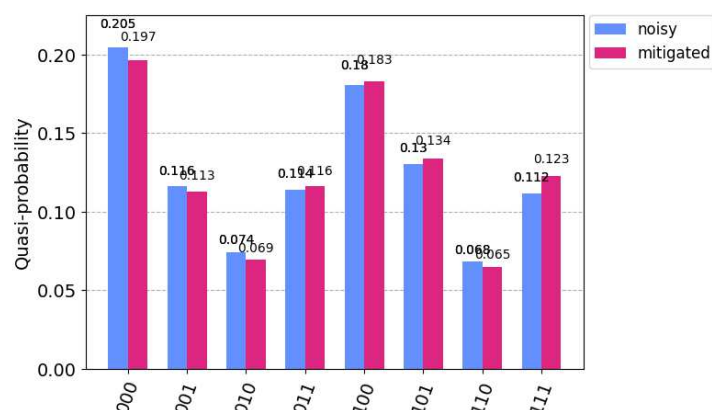
Sont commentés les lignes permettant de choisir un ordinateur quantique réel sur lequel envoyer le circuit, en choisissant celui qui est le moins occupé et qui répond à certains critères. Néanmoins, dans ce cas, nous allons envoyer utiliser les résultats d'un circuit envoyé précédemment.

```

In [17]: 1 from qiskit.utils.mitigation import complete_meas_cal,
          2 CompleteMeasFitter
          3
          4 meas_calibs, state_labels = complete_meas_cal(qubit_list=[0,1,2],
          5                                           circlabel='mcal')
          6
          7 # device = least_busy(
          8 #     provider.backends(
          9 #         filters=lambda x: x.configuration().n_qubits >= 7
          10 #             and not x.configuration().simulator
          11 #             # Not a simulator
          12 #             and x.status().operational == True
          13 #             # Operational backend
          14 #     )
          15 # )
          16 #
          17 # display(device)
          18 #
          19 # Submit a job.
          20 # jobcal = execute(meas_calibs, backend=device)
          21 # job_monitor(jobcal)
          22 # job = execute(qc, backend=device, optimization_level=3, shots=
          23 #     shots)
          24 # job_monitor(job)
          25
          26 # Execute the 31.08.2023 on ibm_nairobi
          27 jobcal = service.job('cjo1ngpd1l3gjfa1lne0')
          28 job = service.job('cjo1svhpthn588jkkphg')
          29
          30 count = job.result().get_counts()
          31 meas_cal = jobcal.result()
          32
          33 meas_fitter = CompleteMeasFitter(meas_cal, state_labels, circlabel='
          34     mcal')
          35 meas_filter = meas_fitter.filter
          36
          37 mitigated_results = meas_filter.apply(job.result())
          38 mitigated_counts = mitigated_results.get_counts()
          39
          40 plot_histogram([count, mitigated_counts], legend=['noisy', '
          41     mitigated'])

```

Out[17]: 1



Nous voyons donc d'où viennent les résultats obtenus dans le chapitre 9. Par cet exemple, cela offre aussi un aperçu de la manière dont les circuits quantiques sont construits et utilisés dans ce document.

Remerciements

Tout ce travail n'aurait pas été possible sans l'aide de nombreuses personnes que je tiens à remercier ici. Un grand merci à mon professeur, M. De Montmollin, pour m'avoir donné l'opportunité de réaliser ce travail de maturité sur un sujet qui m'intéresse et pour m'avoir soutenu tout au long de ce travail, ainsi que de m'avoir aidé à trouver des contacts dans le domaine. Je remercie également M. Lévêque, maître d'enseignement et de recherche à l'EPFL, pour la relecture de ce travail. Je remercie aussi M. Charbon, professeur à l'EPFL et directeur du laboratoire d'architecture quantique, pour l'entretien extrêmement intéressant que j'ai eu avec lui et pour m'avoir apporté des réponses sur de nombreux points.

Je désire également mentionner diverses conférences et séminaires auxquels j'ai assisté et qui m'ont permis d'approfondir mes connaissances sur le sujet. En particulier, la lecture de M. Aspect à l'occasion du Physics Day 2023 à l'EPFL, qui était très intéressante et qui m'a permis de mieux comprendre certains enjeux du domaine des innovations quantiques, ainsi que les différents intervenants lors du Quantum Industry Day in Switzerland 2023, qui m'ont offert de nombreuses perspectives sur le sujet. Dans ce contexte, je tiens aussi à remercier IBM pour les ordinateurs mis à disposition qui m'ont permis d'exécuter les algorithmes quantiques présentés dans ce travail, ainsi que pour leurs ressources éducatives en ligne qui m'ont permis d'apprendre une grande partie de ce que je sais actuellement sur le sujet.

Enfin, je remercie mes parents pour leur soutien et leur relecture de ce travail, ainsi que mon grand-père pour sa relecture attentive des formules physiques et ses conseils avisés, tout particulièrement sur tout ce qui touche à la physique quantique. Finalement, je remercie pêle-mêle tout le reste de ma famille et mes amis pour m'avoir supporté tout au long de ce travail.

Bibliographie

1. WIKIPEDIA CONTRIBUTORS. *Natural computing* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Natural_computing (cf. p. 4).
2. WIKIPEDIA CONTRIBUTORS. *Quantum computing* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_computing (cf. p. 4).
3. *IBM Unveils 400 Qubit-Plus Quantum Processor and Next-Generation IBM Quantum System Two* <https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two> (cf. p. 5).
4. *L'ordinateur quantique est à la croisée des chemins* <https://www.unige.ch/campus/147/dossier6/> (cf. p. 5).
5. *qubit logique* <https://www.culture.fr/franceterme/terme/QUAN10> (cf. p. 5).
6. *What are Q# and the Quantum Development Kit?* <https://learn.microsoft.com/en-us/azure/quantum/overview-what-is-qsharp-and-qdk> (cf. p. 5).
7. *Microsoft Quantum* <https://www.microsoft.com/en-us/quantum> (cf. p. 5).
8. *Site officiel Qiskit* <https://qiskit.org> (cf. p. 5).
9. *IBM Quantum* <https://quantum-computing.ibm.com/> (cf. p. 5).
10. WIKIPEDIA CONTRIBUTORS. *Computational complexity* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Computational_complexity (cf. p. 9).
11. WIKIPEDIA CONTRIBUTORS. *Big O notation* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Big_O_notation (cf. p. 9).
12. *Time Complexity of operation - Python* <https://stackoverflow.com/questions/44020182/time-complexity-of-operation-python> (cf. p. 10).
13. WIKIPEDIA CONTRIBUTORS. *Matrix multiplication algorithm* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Matrix_multiplication_algorithm (cf. p. 11).
14. WIKIPEDIA CONTRIBUTORS. *Algorithme de Coppersmith-Winograd* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Algorithme_de_Coppersmith-Winograd (cf. p. 11).
15. WIKIPEDIA CONTRIBUTORS. *Loi de Moore* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Loi_de_Moore (cf. p. 12).
16. WIKIPEDIA CONTRIBUTORS. *Computational complexity theory* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Computational_complexity_theory (cf. p. 12).
17. WIKIPEDIA CONTRIBUTORS. *BQP* — *Wikipedia, The Free Encyclopedia* 2023. <https://en.wikipedia.org/wiki/BQP> (cf. p. 13).
18. WIKIPEDIA CONTRIBUTORS. *Fentes de Young* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Fentes_de_Young (cf. p. 13).

19. WIKIPEDIA CONTRIBUTORS. *Quantum mechanics* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_mechanics (cf. p. 14).
20. WIKIPEDIA CONTRIBUTORS. *Effet photoélectrique* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Effet_photo%C3%A9lectrique (cf. p. 14).
21. DAMOUR, T. & BURNIAT, M. *le mystère du monde quantique* Dargaud (2016) (cf. p. 14).
22. WIKIPEDIA CONTRIBUTORS. *Onde sur une corde vibrante* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Onde_sur_une_corde_vibrante (cf. p. 15).
23. EXPLAINED, P. *What is the Schrödinger Equation ? A basic introduction to Quantum Mechanics* <https://www.youtube.com/watch?v=2WPA1L9uJqo> (cf. p. 16).
24. WIKIPEDIA CONTRIBUTORS. *Quantum superposition* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_superposition (cf. p. 19).
25. WIKIPEDIA CONTRIBUTORS. *Chat de Schrödinger* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Chat_de_Schr%C3%B6dinger (cf. p. 19).
26. WIKIPEDIA CONTRIBUTORS. *Double-slit experiment* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Double-slit_experiment (cf. p. 19).
27. WIKIPEDIA CONTRIBUTORS. *Quantum entanglement* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_entanglement (cf. p. 20).
28. WIKIPEDIA CONTRIBUTORS. *Expérience d'Aspect* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Exp%C3%A9rience_d%27Aspect (cf. p. 21).
29. WIKIPEDIA CONTRIBUTORS. *Logic gate* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Logic_gate (cf. p. 23).
30. WIKIPEDIA CONTRIBUTORS. *MOSFET* — *Wikipedia, The Free Encyclopedia* 2023. <https://en.wikipedia.org/wiki/MOSFET> (cf. p. 27).
31. WIKIPEDIA CONTRIBUTORS. *Théorie des bandes* — *Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Th%C3%A9orie_des_bandes (cf. p. 28).
32. WIKIPEDIA CONTRIBUTORS. *Quantum logic gate* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_logic_gate (cf. p. 30).
33. WIKIPEDIA CONTRIBUTORS. *Pauli matrices* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Pauli_matrices (cf. p. 30).
34. WIKIPEDIA CONTRIBUTORS. *Bloch sphere* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Bloch_sphere (cf. p. 31).
35. WIKIPEDIA CONTRIBUTORS. *Superdense coding* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Superdense_coding (cf. p. 38).
36. WIKIPEDIA CONTRIBUTORS. *No-cloning theorem* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/No-cloning_theorem (cf. p. 41).
37. WIKIPEDIA CONTRIBUTORS. *Quantum teleportation* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_teleportation (cf. p. 42).
38. DEUTSCH, D., BARENCO, A. & EKERT, A. Universality in Quantum Computation. <https://arxiv.org/abs/quant-ph/9505018> (1995) (cf. p. 44).
39. VESELY, M. *How to approximate Rx, Ry and Rz gates ?* <https://quantumcomputing.stackexchange.com/questions/11861/how-to-approximate-rx-ry-and-rz-gates> (cf. p. 44).
40. BUDINGER, N., FURUSAWA, A. & van LOOCK, P. *All-optical quantum computing using cubic phase gates* 2022. <https://arxiv.org/abs/2211.09060> (cf. p. 44).

41. SAJEED, S., AHMED, A., ULLAH, S. M. & MOZUMDER, Z. H. *An approach to realize a quantum Hadamard gate through optical implementation in 2010 IEEE International Conference on Electro/Information Technology* (IEEE, mai 2010). <https://doi.org/10.1109/eit.2010.5612120> (cf. p. 44).
42. ROSENBLUM, S., GAO, Y. Y., REINHOLD, P., WANG, C., AXLINE, C. J., FRUNZIO, L. *et al.* A CNOT gate between multiphoton qubits encoded in two cavities. *Nature Communications* **9**. <https://doi.org/10.1038/s41467-018-03059-5> (fév. 2018) (cf. p. 45).
43. NOH, T., PARK, G., LEE, S.-G., SONG, W. & CHONG, Y. Construction of controlled-NOT gate based on microwave-activated phase (MAP) gate in two transmon system. *Scientific Reports* **8**. <https://doi.org/10.1038/s41598-018-31896-3> (sept. 2018) (cf. p. 45).
44. TROUSDALE, J. *How can we implement controlled-T gate using CNOT and H, S and T gates ?* <https://quantumcomputing.stackexchange.com/questions/13132/how-can-we-implement-controlled-t-gate-using-cnot-and-h-s-and-t-gates> (cf. p. 45).
45. WIKIPEDIA CONTRIBUTORS. *IBM Quantum Experience — Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/IBM_Quantum_Experience (cf. p. 45).
46. WIKIPEDIA CONTRIBUTORS. *Superconducting quantum computing — Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Superconducting_quantum_computing (cf. p. 45).
47. WIKIPEDIA CONTRIBUTORS. *Charge qubit — Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Charge_qubit (cf. p. 45).
48. WIKIPEDIA CONTRIBUTORS. *Trapped ion quantum computer — Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Trapped_ion_quantum_computer (cf. p. 46).
49. WIKIPEDIA CONTRIBUTORS. *Transmon — Wikipedia, The Free Encyclopedia* 2023. <https://en.wikipedia.org/wiki/Transmon> (cf. p. 46).
50. WIKIPEDIA CONTRIBUTORS. *Josephson effect — Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Josephson_effect (cf. p. 46).
51. WIKIPEDIA CONTRIBUTORS. *Cooper pair — Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Cooper_pair (cf. p. 46).
52. GIDNEY, C. *How are the IBM's and Google's Hadamard gates fabricated and operated ?* <https://quantumcomputing.stackexchange.com/questions/14576/how-are-the-ibms-and-googles-hadamard-gates-fabricated-and-operated> (cf. p. 46).
53. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A : Mathematical and Physical Sciences* **439**, 553-558. <https://doi.org/10.1098/rspa.1992.0167> (déc. 1992) (cf. p. 50).
54. CLEVE, R., EKERT, A., MACCHIAVELLO, C. & MOSCA, M. Quantum algorithms revisited. *Proceedings of the Royal Society of London. Series A : Mathematical, Physical and Engineering Sciences* **454**, 339-354. <https://doi.org/10.1098/rspa.1998.0164> (jan. 1998) (cf. p. 50).
55. WIKIPEDIA CONTRIBUTORS. *Deutsch-Jozsa algorithm — Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Deutsch%E2%80%93Jozsa_algorithm (cf. p. 50).
56. WIKIPEDIA CONTRIBUTORS. *Série de Fourier — Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/S%C3%A9rie_de_Fourier (cf. p. 55).
57. WIKIPEDIA CONTRIBUTORS. *Transformation de Fourier discrète — Wikipedia, The Free Encyclopedia* 2023. https://fr.wikipedia.org/wiki/Transformation_de_Fourier_discr%C3%A8te (cf. p. 55).

58. WIKIPEDIA CONTRIBUTORS. *Quantum Fourier transform* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_Fourier_transform (cf. p. 55).
59. WIKIPEDIA CONTRIBUTORS. *Quantum phase estimation algorithm* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_phase_estimation_algorithm (cf. p. 57).
60. SHOR, P. *Algorithms for quantum computation : discrete logarithms and factoring* in *Proceedings 35th Annual Symposium on Foundations of Computer Science* (IEEE Comput. Soc. Press). <https://doi.org/10.1109/sfcs.1994.365700> (cf. p. 61).
61. WIKIPEDIA CONTRIBUTORS. *Integer factorization* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Integer_factorization (cf. p. 61).
62. WIKIPEDIA CONTRIBUTORS. *Shor's algorithm* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Shor%27s_algorithm (cf. p. 61).
63. WIKIPEDIA CONTRIBUTORS. *RSA (cryptosystem)* — *Wikipedia, The Free Encyclopedia* 2023. [https://en.wikipedia.org/wiki/RSA_\(cryptosystem\)](https://en.wikipedia.org/wiki/RSA_(cryptosystem)) (cf. p. 65).
64. A-TA-V, ANIS, M. S., ABBY-MITCHELL, ABRAHAM, H., ADUOFFEI, AGARWAL, R. *et al.* *Qiskit : An Open-source Framework for Quantum Computing* 2021 (cf. p. 66).
65. BEAUREGARD, S. Circuit for Shor's algorithm using $2n+3$ qubits. <https://arxiv.org/abs/quant-ph/0205095> (2002) (cf. p. 66).
66. WIKIPEDIA CONTRIBUTORS. *Grover's algorithm* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Grover%27s_algorithm (cf. p. 70).
67. GROVER, L. K. *A fast quantum mechanical algorithm for database search* in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96* (ACM Press, 1996). <https://doi.org/10.1145/237814.237866> (cf. p. 70).
68. WIKIPEDIA CONTRIBUTORS. *Quantum counting algorithm* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_counting_algorithm (cf. p. 74).
69. SMITH, J. G., BARNES, C. H. W. & ARVIDSSON-SHUKUR, D. R. M. *An iterative quantum-phase-estimation protocol for near-term quantum hardware* 2022. <https://arxiv.org/abs/2206.06392> (cf. p. 79).
70. FEYNMAN, R., LEIGHTON, R. B. & SANDS, M. L. in *The Feynman lectures on physics* 1-9 (1963-1965) (cf. p. 84).
71. PATRA, B., INCANDELA, R. M., van DIJK, J. P. G., HOMULLE, H. A. R., SONG, L., SHAHMOHAMMADI, M. *et al.* Cryo-CMOS Circuits and Systems for Quantum Computing Applications. *IEEE Journal of Solid-State Circuits* **53**, 309-321 (2018) (cf. p. 86).
72. WIKIPEDIA CONTRIBUTORS. *Quantum error correction* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_error_correction (cf. p. 86).
73. *ID Quantique* <https://www.idquantique.com> (cf. p. 87).
74. LIAO, S.-K., CAI, W.-Q., LIU, W.-Y., ZHANG, L., LI, Y., REN, J.-G. *et al.* Satellite-to-ground quantum key distribution. *Nature* **549**, 43-47. <https://doi.org/10.1038/nature23655> (août 2017) (cf. p. 87).
75. *IBM welcomes CERN as a new hub in the IBM Quantum Network* <https://research.ibm.com/blog/cern-lhc-qml> (cf. p. 88).
76. BROWN, A. R. & SUSSKIND, L. A holographic wormhole traversed in a quantum computer. *Nature* **612**, 41-42. <https://doi.org/10.1038/d41586-022-03832-z> (nov. 2022) (cf. p. 88).

77. ROMERO, J. & ASPURU-GUZZI, A. *Variational quantum generators : Generative adversarial quantum machine learning for continuous distributions* 2019. <https://arxiv.org/abs/1901.00848> (cf. p. 88).
78. WIKIPEDIA CONTRIBUTORS. *Quantum complexity theory* — *Wikipedia, The Free Encyclopedia* 2023. https://en.wikipedia.org/wiki/Quantum_complexity_theory (cf. p. 88).
79. ANTONI-MICOLIER, L., CARBONE, D., MÉNORET, V., LAUTIER-GAUD, J., KING, T., GRECO, F. *et al.* Detecting Volcano-Related Underground Mass Changes With a Quantum Gravimeter. *Geophysical Research Letters* **49**. <https://doi.org/10.1029/2022gl097814> (juin 2022) (cf. p. 88).
80. WOOTTON, J. R., HARKINS, F., BRONN, N. T., VAZQUEZ, A. C., PHAN, A. & ASFAW, A. T. *Teaching quantum computing with an interactive textbook* in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)* (2021), 385-391 (cf. p. 90).