Master Systèmes Dynamiques et Signaux

Rapport Bibliographique

Informatique Quantique

Auteur :
M. Pierre Engelstein

Encadrants :
Dr. Nicolas Delanoue
Pr. François Chapeau-Blondeau

Jury :
Pr. Laurent Hardouin
Dr. Nicolas Delanoue
Pr. François Chapeau-Blondeau
Pr. Sébastien Lahaye
Dr. Mehdi Lhommeau
Dr. Remy Guyonneau

Version du 21 janvier 2021

Remerciements

Table des matières

Table des figures

Chapitre 1

Introduction

Chapitre 2

Informatique quantique : éléments de base

2.1 Postulats de base

On pose 3 postulats, servant de base aux raisonnements qui suivront. Ces postulats sont des hypothèses de travail, mais confirmés jusqu'a présent par les expériences.

2.1.1 Etat d'un système quantique

Un système quantique peut être représenté par un vecteur d'état, de la même manière qu'un système physique classique. On le représente par la notation de Dirac, notée de la forme $|\psi\rangle$. Ce vecteur d'état est necessairement de norme 1 (la somme des modules au carré vaut 1). On peut distinguer deux types d'états pour un système quantique : les états de base, ou **états purs**, formant une base vectorielle, et les états superposés. Ces états superposés correspondent à une combinaison linéaire des états purs. On peut écrire généralement un état quantique de la façon suivante :

$$|\psi\rangle = \sum_i c_i \, |k_i\rangle \, , \tag{2.1}$$

avec les $|k_i\rangle$ états purs, et les c_i respectant $\sum_i |c_i|^2 = 1$ pour la normalité du vecteur d'état.

Dans le cadre de l'informatique quantique, on utilise le système quantique le plus simple, appelé **qubit**. Ce système quantique est composé de deux états purs, $|0\rangle$ et $|1\rangle$, et des états superposés. Similairement à l'informatique classique, où on travaille sur le système physique le plus élémentaire - le bit -

en quantique on travaille sur le système physique quantique élémentaire - le qubit. On dispose des mêmes états purs, mais l'informatique quantique apporte les états *intermédiaires* superposés. Dans la base canonique $\{|0\rangle, |1\rangle\}$, on note un qubit de la façon suivante : $|\psi\rangle = \alpha \cdot |0\rangle + \beta \cdot |1\rangle$.

2.1.2 Mesure projective

Que se passe-t-il quand on mesure un système quantique ? On a évoqué au dessus la notion de superposition des états. L'expérience montre que, lorsqu'on va mesurer un système quantique, on va mesurer au hasard un des états de base, avec comme probabilité le carré du coefficient correspondant.

Mathématiquement, la mesure effectue une projection de l'état du système sur un des états de base dont il est composé. Par exemple, si on a un qubit dans l'état $|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$, alors la probabilité de mesurer 0, c'est à dire de projeter le système dans l'état pur $|0\rangle$ est $|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$; et de même pour l'état pur $|1\rangle$. On a donc exactement la même probabilité de mesurer 0 que de mesurer 1.

Il faut noter que, lorsqu'on fait la mesure, on projete réellement le système quantique dans l'état pur. Concrètement, si on a un état superposé qu'on mesure, il se place dans l'état pur qu'on mesure, et toutes les mesures successives qu'on fera sur ce qubit donneront le même résultat. La mesure fait donc perdre l'état qu'on avait auparavant.

2.1.3 Dynamique du système

Comme n'importe quel système physique, on peut faire évoluer un système quantique dans le temps. Ici apparaissent deux propriétés. Tout d'abord, il découle du premier postulat que la dynamique d'un système quantique est necessairement unitaire. En effet, un système quantique doit, pour être valide, avoir une norme de 1, et donc l'évolution d'un système quantique d'un premier état vers un autre doit conserver cette unitarité. Cela veut dire que la matrice représentant l'évolution du système quantique doit respecter la propriété suivante :

$$U^\dagger U = U U^\dagger = I, \tag{2.2}$$

avec U la matrice d'évolution du système, $U^\dagger$ la matrice conjuguée transposée de U, et I l'identité.

Une deuxième propriété est également posée, ne découlant pas des deux postulats précédents. La dynamique d'un système quantique doit être aussi linéaire. Ainsi, on pourrait penser que n'importe quelle évolution unitaire

serait valable, mais l'expérience nous montre que non, il faut en plus qu'elle soit linéaire.

2.2 Vers une informatique quantique

A partir de ces 3 postulats de base, on peut commencer à comprendre comment se construit l'informatique quantique, et quels sont les apports sur l'informatique classique.

2.2.1 Multiples qubits

On a vu la définition d'un qubit. Cela nous permet d'étendre ce système quantique élémentaire à des systèmes composés de multiples qubits. En informatique classique, on travaille quasi systématiquement sur des mots binaires plutôt que des bits uniques ; l'équivalent est vrai en quantique. Pour cela, les systèmes quantiques, et donc les qubits, sont munis d'une opération : le produit tensoriel. Quand on veut effectuer une combinaison de deux qubits, cela revient à faire un produit tensoriel des états des deux qubits individuels. Par exemple, si nous disposons de deux qubits ayant pour valeur $|\psi_1\rangle = |0\rangle$ et $|\psi_2\rangle = |1\rangle$, alors on peut écrire le 2-qubit combinaison des deux de la façon suivante :

$$|\psi\rangle = |0\rangle \otimes |1\rangle \,, \tag{2.3}$$

qu'on écrit généralement sous la forme plus simple :

$$|\psi\rangle = |01\rangle \,. \tag{2.4}$$

Prenons un 2-qubit formé par la combinaison de 2 qubits :

$$\begin{aligned} |\psi\rangle &= (\alpha_1 \cdot |0\rangle + \beta_1 \cdot |1\rangle) \otimes (\alpha_2 \cdot |0\rangle + \beta_2 \cdot |1\rangle) \\ &= \alpha_1\alpha_2 \,|0\rangle \otimes |0\rangle + \alpha_1\beta_2 \,|0\rangle \otimes |1\rangle + \beta_1\alpha_2 \,|1\rangle \otimes |0\rangle + \beta_1\beta_2 \,|1\rangle \otimes |1\rangle \\ &= \gamma_1 \,|00\rangle + \gamma_2 \,|01\rangle + \gamma_3 \,|10\rangle + \gamma_4 \,|11\rangle \end{aligned}$$

On peut donc, si on a un 2-qubit combinaison linéaire de tout les états de bases, le séparer en deux qubits individuels, sur lesquels on va pouvoir agir.

Considérons maintenant le 2-qubit suivant :

$$|\psi\rangle = \gamma_1 \,|00\rangle + \gamma_2 \,|11\rangle$$

Il parait évident alors qu'on ne peut pas séparer ce 2-qubit en produit tensoriel de 2 qubits individuels. Dans ce cas, on dit que les deux qubits sont **intriqués** et donc non séparables.

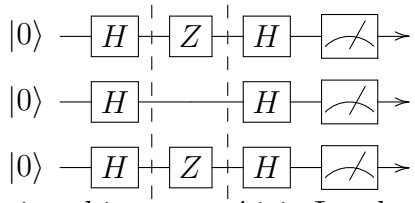
2.2.2 Portes quantiques

Dans la représentation d'état classique, et spécifiquement en informatique, on peut faire évoluer l'état au travers de portes. En informatique classique, on dispose ainsi de portes élémentaires telles que `AND`, `NOT`, `OR`, etc.

De la même manière, en respectant le troisième postulat posé précédement, on peut construire des portes logiques quantiques, les combiner, afin de créer des circuits quantiques. En informatique quantique, on distingue donc plusieurs portes élémentaires, utilisés dans beaucoup de circuit :

1. La porte de Hadamard H. Elle permet de passer un qubit d'un état pur $|0\rangle$ à l'état superposé $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$, ou de l'état pur $|1\rangle$ à l'état superposé $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$. Elle est très utilisée en début de circuit pour préparer les qubits entrants dans un état permettant l'évaluation parallèle de toutes les entrées ;
2. Les portes de Pauli X, Y et Z permettant d'effectuer des rotations aux états des qubits ;
3. La porte de Toffoli, équivalent d'un `NON` booléen, est une porte universelle quantique. Elle permet donc de construire l'ensemble des autres portes faisables.

Un exemple de circuit est le suivant : On dispose d'un 3-qubit dans l'état $|000\rangle$. Au départ, on applique à ces trois qubits une porte de Hadamard, qui les fait se retrouver dans des états équilibrés. On arrive donc à un 3-qubit qui est équilibré entre chacun de ses états purs. On applique ensuite deux portes de Pauli Z, un au premier qubit, et un au troisième. Cela est effectivement faisable puisqu'a la sortie des portes de hadamard, les 3 qubits ne sont pas intriqués. On applique de la même façon 3 portes de Hadamard à la sortie, puis on mesure.



L'avantage du quantique bien montré ici : Les deux portes du milieu vont faire changer l'état des qubits, mais en parallèle : on fait évoluer le système simultanément pour tout les états purs qui nous intéressent, puisqu'ils sont superposés.

Chapitre 3

Algorithmes quantiques

3.1 Algorithme de Deutsch-Jozsa

Le problème de Deutsch-Jozsa s'intéresse à une fonction booléenne qui est soit équilibrée (ayant autant d'entrées à 1 qu'a 0), soit constante (toutes les entrées à 1, ou toutes les entrées à 0).

Problème 1 (Deutsch-Jozsa)**.** *Etant donnée une fonction f booléenne qui est soit équilibrée, soit constante. Le problème de Deutsch-Jozsa est de déterminer si f est constante ou équilibrée.*

Dans le cas classique, il faut effectuer au pire $2^{n-1}+1$ évaluations pour déterminer si f est constante ou équilibrée. Tout d'abord, dès que deux évaluations sont différentes, f est nécessairement équilibrée. De plus, si après avoir évalué 2^{n-1} entrées et obtenu la même valeur, une évaluation supplémentaire nous permet de connaitre dans quelle catégorie f se trouve.

Dans le cas quantique, on peut résoudre ce problème en effectuant qu'une seule évaluation, parallèle, de f. Le schéma suivant illustre le fonctionnement de l'algorithme présente par Deutsch et Jozsa en 1992 [1] :

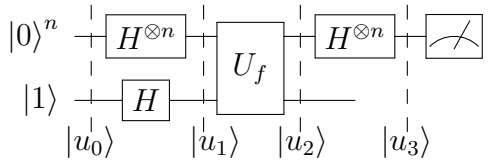


FIGURE 3.1 – Schéma de l'algorithme

Les détails algébriques de l'algorithme sont disponibles en annexes.

On dispose au départ un (n+1)-qubit, c'est à dire (n+1) qubits associés, les n premiers mis à 0 et le dernier mis à 1. Cet état d'entrée $|u_0\rangle$ est construit

préalablement, puis est donné en entrée à des portes de Hadamard. Pour rappel, cette porte permet de faire passer un qubit d'un état pur ($|0\rangle$ ou $|1\rangle$) à un état équilibré.

Au sortir de cette porte, on obtient l'état $|u_1\rangle$, qui si mesuré va nous donner avec autant de probabilité un des états purs. On le passe donc à la "boite" U_f. On appelle ce bloc **oracle**, qui implémente la fonction f qu'on souhaite évaluer pour n qubits (on ne détaille pas ici la façon d'implémenter f). L'entrée étant un (n+1)-qubit équilibré, l'oracle va évaluer en simultané toutes les entrées, correspondant aux états purs superposés. C'est ici qu'on peut à nouveau montrer la force de l'informatique quantique : avec en entrée un état superposé, un bloc fonctionnel va faire évoluer le système en effectuant le calcul pour tout les états purs superposés composant l'entrée.

On obtient $|u_2\rangle$, qui contient la solution de notre problème. Néanmoins, si on essaye ici de mesurer le système, on ne va pas obtenir directement une réponse "oui" ou "non" à notre problème : le système est encore dans un état superposé, et la réponse est codée dans les modules associés aux états purs superposés. Si on mesure $|u_2\rangle$, on obtiendra aléatoirement un des état purs composant le système.

Pour résoudre ce problème, on applique à nouveau une porte de Hadamard pour obtenir un $|u_3\rangle$. On l'a dit précédement, quand appliqué à un état pur, la porte de Hadamard permet de créer un état équilibré. En revanche, quand comme ici on l'applique à un état déjà superposé, on peut obtenir directement un état qui va nous donner à coup sûr une réponse binaire. Ici, quand on va mesurer $|u_3\rangle$, on va soit mesurer un mot binaire de n bits valant 0, et dans ce cas la fonction est constante. Si on obtient n'importe quelle autre solution, alors la fonction est équilibrée. A nouveau, les détails algébriques de cet algorithme sont disponibles en annexes.

Cet algorithme permet principalement de prouver la supériorité de l'informatique quantique sur ce type de problèmes : on arrive effectivement à atteindre des performances qui sont inatteignables avec des ordinateurs classiques. Plusieurs autres algorithmes ont été créés en se basant sur celui-là, en modifiant le problème et la fonction évaluée, mais en gardant l'algorithme. On peut notament citer l'algorithme de Bernstein-Vazirani [2] qui permet de résoudre un problème d'arithmétique modulaire.

3.2 Algorithme de Grover

On s'intéresse maintenant à un problème classique d'informatique : la recherche dans des listes. En informatique classique, on connait un certain

nombre d'algorithmes pour effectuer des recherches dans des listes, qu'elles soient triées ou non triées.

Problème 2 (Grover)**.** *Soit une liste non triée à N entrées. Le problème de Grover est de rechercher une entrée spécifique dans cette liste de façon efficace.*

Classiquement, l'algorithme le plus efficace devra effectuer au pire N itérations sur la liste pour trouver l'élément voulu. La liste étant non triée, on ne peut pas utiliser des algorithmes qui divisent la liste, qui permettent d'obtenir des performances en $\mathcal{O}(\log N)$.

En utilisant les principes de l'informatique quantique, Lou Grover a proposé en 1992 [3] un algorithme permettant de trouver la solution avec une forte probabilité, en juste $\mathcal{O}(\sqrt{N})$

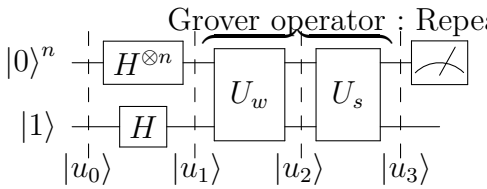


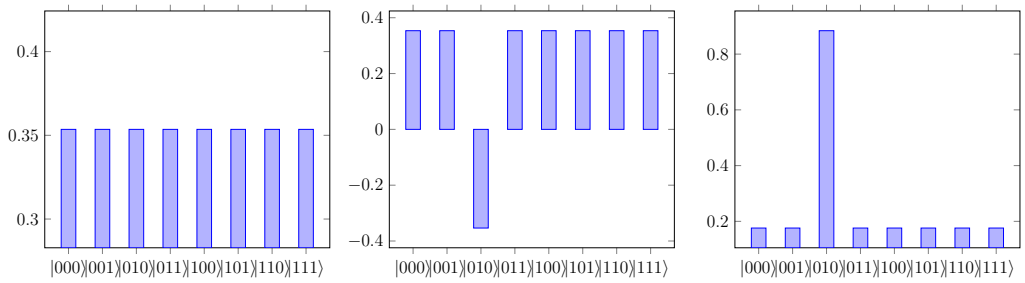
FIGURE 3.2 – Schéma de l'algorithme

De même que pour Deutsch-Jozsa, les détails algébriques de cet algorithme sont disponibles en annexes.

On commence donc de la même manière avec n qubits initialisés à $|0\rangle$ et 1 à $|1\rangle$, et on les passe tous dans une porte de Hadamard pour avoir des qubits équilibrés, afin de faire un traitement parallèle.

La deuxième étape consiste à appliquer l'opérateur de Grover à l'état superposé $|u1\rangle$. On vient effectuer deux opérations successives ici.

1. Une opération d'inversion d'amplitude avec l'opérateur U_w. On a au départ les n états purs superposés de façon équilibrés, ils ont donc tous la même amplitude (le module devant l'état pur). Cet opérateur vient inverser l'amplitude de l'état pur qu'on cible. On se retrouve après dans un état équilibré, puisque les probabilités sont les amplitudes mises au carrés. On a donc la réponse dans l'état quantique, mais elle est inaccessible à la mesure.
2. Une opération de miroir à la moyenne avec l'opérateur U_s. Cet opérateur vient calculer la moyenne des amplitudes, puis enlever aux amplitudes leur différence à la moyenne. Dans le cas des amplitudes qui n'ont pas bougé avec U_w, on se retrouve légèrement en dessous de la moyenne, et dans le cas de l'état ciblé, on se retrouve bien au dessus de la moyenne.



(a) $|u_1\rangle$ (b) $|u_2\rangle$ (c) $|u_3\rangle$

FIGURE 3.3 – Evolution des amplitudes pour $n = 3$ qubits

Cette figure illustre cette évolution pour 3 qubits. On voit qu'a la fin de U_s, on obtient un état quantique qui a une forte probabilité de tomber sur l'état voulu (ici, $|010\rangle$), les amplitudes de ceux qu'on ne veut pas ayant été fortement réduites. Néanmoins, on est toujours pas sûr de tomber sur le bon résultat : dans ce cas précis, la probabilité de tomber sur $|010\rangle$ est d'environ 78%, on a quand même quasiment 20% de chances de mesurer un autre état. Ce n'est pas satisfaisant.

La solution est de réappliquer les portes U_w et U_s. Ceci va encore plus diminuer les probabilités des états non voulus et augmenter la probabilité de l'état voulu. On peut prouver, notament géométriquement, que le nombre d'optimal d'itérations est $\frac{\pi}{4}\sqrt{N}$. En fait, il se passe autre chose : si on dépasse ce nombre optimal d'itération, on va diminuer l'amplitude de l'état voulu et augmenter les probabilités des autres états : c'est cyclique. La figure suivante illustre une simulation sur plus d'itérations que ce qui est necessaire :

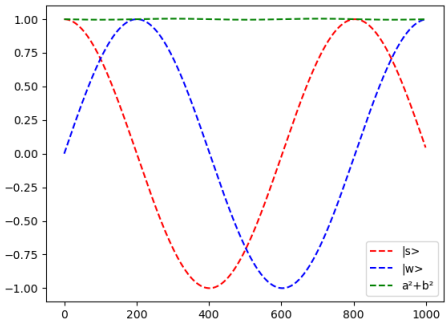


FIGURE 3.4 – Evolution des amplitudes pour n=16, sur 1000 itérations

En bleu, on représente la probabilité de tomber sur l'état cible, en rouge la probabilité de tomber sur un autre état (en vert, pour indication, la somme

des probabilités pour vérifier que la somme vaut bien 1). On voit bien ici le rebouclement après $\frac{\pi}{4}\sqrt{N}$ itérations.

3.3 Algorithme de Shor

On peut enfin étudier l'algorithme de Shor, proposé par Peter Shor en 1997 [4].

Problème 3. *On s'intéresse au problème de la factorisation d'un nombre $N = q \times q$ avec p et q nombres entiers très grands.*

Classiquement, ce problème est résolvable en $\mathcal{O}(\exp[c.n^{\frac{1}{3}}(\log n)^{\frac{2}{3}}])$ avec l'algorithme du crible du corps algébrique. Cet algorithme est certes efficace pour des nombres supérieurs à 10^{100}, mais ne permet pas de factoriser pour des nombres de 2048 ou 4096 bits. Ce domaine de recherche avance rapidement, et de nouveaux algorithmes sont trouvés permettant d'améliorer les performances. On ne trouve en revanche toujours pas d'algorithmes permettant de résoudre ce problème en un temps polynomial.

En utilisant l'informatique, Shor a démontré que ce problème était résolvable bien plus rapidement avec un processeur quantique, en un peu plus rapidement que $\mathcal{O}(n^3)$. En fait, Shor résout avec son algorithme une sous-partie d'un algorithme :

```
Data: N
repeat
    choose a coprime with N ;
    find smallest r such that a^r ≡ 1(modN) ;
    if r is even then
        x ≡ a^(r/2)(modN) ;
        if x + 1 ≢ 0(modN) then
            at least one of {p, q} ∈ {gcd(x + 1, N), gcd(x − 1, N)} ;
            break ;
        else
            continue ;
        end
    else
        continue ;
    end
until;
```

Cet algorithme permet de trouver les facteurs premiers p et q. On peut prouver que la deuxième instruction revient à chercher la période de la fonc-

tion $a^r \pmod N$. C'est cette instruction spécifique qui est compliquée à résoudre classiquement, mais que Shor à réussi à résoudre en quantique.

Cet algorithme se passe en 3 étapes. Les deux premières sont similaires aux deux algorithmes précédents : on fait passer un n-qubit de l'état $|0\rangle^{\otimes n}$ à l'état équilibré ; puis on évalue la fonction $a^r \pmod N$ pour l'état équilibré. La troisième étape fait intervenir une nouvelle notion : la transformée de Fourier quantique. C'est une généralisation de la porte de Hadamard, qui fait passer les qubits de la base $\{|0\rangle, |1\rangle\}$ à la base $\{|+\rangle, |-\rangle\}$ (pour rappel, on a $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, et $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$).

Annexe A

Algorithme de Deutsch-Jozsa

A.1 Problème à résoudre

Soit une fonction f booléenne définie par

$$\begin{array}{rccl} f: & \{0,1\}^n & \to & \{0,1\} \\ & (x_0, x_1, \ldots, x_n) & \mapsto & y = f(x_0, x_1, \ldots, x_n), \end{array}$$

Définition 1. *Une fonction booléenne f est dite équilibrée si f retourne 0 pour la moitié de ses entrées.*

Définition 2. *Une fonction est dite constante si elle retourne une constante pour toutes ses entrées.*

Remarque 1. *Avec n un entier et comme les fonctions booléennes sont à valeur dans $\{0,1\}$, il n'existe que deux fonctions constantes f_0 et f_1.*

Exemple 1. *Soit f la fonction booléenne $f : \{0,1\}^2 \to \{0,1\}$ définie par la table de vérité suivante :*

(x_1, x_2)	$f(x_1, x_2)$
$(0,0)$	0
$(0,1)$	1
$(1,0)$	1
$(1,1)$	0

Cette fonction est équilibrée. On notera qu'elle correspond au classique "ou exclusif". Cette fonction pourrait être représentée par le vecteur de ces valeurs : $(0,1,1,0)$. Elle peut aussi être codée en listant les entrées où elle est vraie, ici $\{(0,1),(1,0)\}$ (ou bien en base 10 : $\{1,2\}$).

Remarque 2. *Dénombrer les fonctions équilibrées revient à dénombrer les façons de placer le symbole le symbole* 1 *dans la moitié des cases d'un vecteur de taille* 2^n. *Avec les emplacements, cela revient à dénombrer l'ensemble de sous-ensembles de* $\{0, \ldots, 2^n - 1\}$ *qui ont pour cardinal* $2^n/2$. *Finalement, il a donc* $\binom{2^n}{2^{n-1}}$ *fonctions équilibrées. Ce qui fait*

$$\binom{2^n}{2^{n-1}} = \frac{(2^n)!}{(2^{n-1})!(2^n - 2^{n-1})!} = \frac{(2^n)!}{((2^{n-1})!)^2}.$$

On notera que la proportion des fonctions équilibrées sur l'ensemble des 2^{2^n} *fonctions booléennes de* n *variables diminue très rapidement avec* n.

Initialisation

On commence avec : $|u_0\rangle = (|0\rangle^{\bigotimes n}) \otimes |1\rangle$: n-qubits à $|0\rangle$ et 1-qubit à $|1\rangle$.

Etape 1

On applique une porte de Hadamard à $|u_0\rangle$ pour avoir un état équiprobable : $|u_1\rangle = H\,|u_0\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle\,(|0\rangle - |1\rangle)$.

Etape 2

On applique l'oracle quantique suivant à $|u_1\rangle$:

$$o : |x\rangle\,|y\rangle \mapsto |x\rangle\,|y \oplus f(x)\rangle\,.$$

Posons x, on est alors dans l'une des deux situations disjointes suivantes :
— $f(x) = 0$,
— $f(x) = 1$.
Analysons chacune de ces situations, tout d'abord si $f(x) = 0$ alors

$$o : |x\rangle\,(|0\rangle - |1\rangle) \mapsto |x\rangle\,(|0\rangle - |1\rangle).$$

Autrement dit $|x\rangle\,(|0\rangle - |1\rangle$ est un point fixe de o.

Dans l'autre situation, on a $f(x) = 1$ et on en déduit

$$o : |x\rangle\,(|0\rangle - |1\rangle) \mapsto |x\rangle\,(|1\rangle - |0\rangle).$$

Autrement dit, dans ce cas, le vecteur $|x\rangle\,(|0\rangle - |1\rangle)$ est envoyé sur son opposé via o.

Finalement, les deux cas précédents peuvent être résumé sous la forme suivante

$$o : |x\rangle (|0\rangle - |1\rangle) \mapsto (-1)^{f(x)} |x\rangle (|0\rangle - |1\rangle).$$

Par linéarité, on en déduit :

$$|u_2\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle (|0\rangle - |1\rangle). \tag{A.1}$$

On peut ignorer le dernier qubit $(|0\rangle - |1\rangle)$ comme il est constant. Finalement, on en déduit :

$$|u_2\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle . \tag{A.2}$$

Etape 3

Maintenant qu'on a appliqué notre oracle, on est toujours dans un état "probabiliste", et en mesurant nous n'obtiendrons pas une réponse exacte à notre problème. L'objectif est donc maintenant de ramener les solutions sur un état déterminé pour obtenir la réponse systématique. En appliquant la porte de Hadamard, on va pouvoir forcer un état à apparaître pour un type de fonction f, et le forcer à disparaître dans l'autre cas, ce qui nous permet d'avoir une réponse systématique sur le type de la fonction : est-elle équilibrée ou bien constante ?

On réapplique une porte Hadamard à chaque qubit sortant, ce qui donne :

$$|u_3\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \left(\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} (-1)^{x.y} |y\rangle \right).$$

Par linéarité, on a :

$$|u_3\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{y=0}^{2^n-1} (-1)^{f(x)} (-1)^{x.y} |y\rangle . \tag{A.3}$$

La probabilité $|p|$ de mesurer $|0\rangle^{\bigotimes n}$ est donc :

$$|\frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)}|, \tag{A.4}$$

avec $p = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)}$.

Si on a une fonction $f(x)$ constante, alors chaque élément de la somme retourne la même valeur (1 ou -1 suivant que $f(x)$ retourne 0 ou 1), la somme va donc valoir $\pm 2^n$. Dans le cas où la fonction est équilibrée, on va avoir autant de 1 que de -1, la somme est donc nulle.

On a donc les valeurs suivantes dépendant du type de $f(x)$:

1. Si $f(x)$ est constante : $p = \pm\frac{1}{2^n} \times 2^n = \pm 1$,
2. Si $f(x)$ est équilibrée : $p = \pm\frac{1}{2^n} \times 0 = 0$.

Dans le cas constant, on ne peut donc que mesurer $|0\rangle^{\bigotimes n}$ puisqu'il a une probabilité de 1 d'apparaître. Dans le cas équilibré, on ne mesure jamais $|0\rangle^{\bigotimes n}$ puisque sa probabilité est nulle.

On en conclut que, lorsqu'on effectue une mesure, si on tombe sur $|0\rangle^{\bigotimes n}$ alors la fonction est constante, sinon elle est équilibrée.

A.1.1 Exemple

Prenons une fonction f comme définie précédemment avec $n = 2$, sans savoir si elle est constante ou équilibrée.

Etape 1

On commence avec $|u_0\rangle = |001\rangle$. La première étape est l'application de la porte d'hadamard à $|u_0\rangle$:

$$\begin{aligned}
|u_1\rangle &= H|u_0\rangle = H|0\rangle \otimes H|0\rangle \otimes H|1\rangle, \\
&= \frac{1}{2\sqrt{2}}((|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle) \otimes (|0\rangle - |1\rangle)), \\
&= \frac{1}{2\sqrt{2}}\{|000\rangle - |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle + |110\rangle - |111\rangle\}, \\
&= \frac{1}{2\sqrt{2}}\{|00\rangle(|0\rangle - |1\rangle) + |01\rangle(|0\rangle - |1\rangle) + |10\rangle(|0\rangle - |1\rangle) + |11\rangle(|0\rangle - |1\rangle)\}.
\end{aligned} \tag{A.5}$$

Etape 2 : oracle quantique

On applique à $|u_1\rangle$ l'oracle quantique $|x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle$:

$$|u_2\rangle = \frac{1}{2\sqrt{2}} |00\rangle \left(|0 \oplus f(00)\rangle - |1 \oplus f(00)\rangle\right) + |01\rangle \left(|0 \oplus f(01)\rangle - |1 \oplus f(01)\rangle\right) + |10\rangle \left(|0 \oplus f(10)\rangle - |1 \oplus f(10)\rangle\right) + |11\rangle \left(|0 \oplus f(11)\rangle - |1 \oplus f(11)\rangle\right).$$

On peut alors réécrire l'équation de la façon suivante :

$$|u_2\rangle = \frac{1}{2\sqrt{2}} (-1)^{f(00)} |00\rangle \left(|0\rangle - |1\rangle\right) + (-1)^{f(01)} |01\rangle \left(|0\rangle - |1\rangle\right) + (-1)^{f(10)} |10\rangle \left(|0\rangle - |1\rangle\right) + (-1)^{f(11)} |11\rangle \left(|0\rangle - |1\rangle\right).$$

Par la suite, on va appliquer une porte de Hadamard à $|u_2\rangle$. Le qubit $|0\rangle - |1\rangle$ donne $|1\rangle$ par la cette porte, il est donc constant par rapport à $|u_0\rangle$. On peut donc le retirer de l'équation, ce qui nous donne pour $|u_2\rangle$:

$$|u_2\rangle = \frac{1}{2\sqrt{2}} \left((-1)^{f(00)} |00\rangle + (-1)^{f(01)} |01\rangle + (-1)^{f(10)} |10\rangle + (-1)^{f(11)} |11\rangle\right). \tag{A.6}$$

Matriciellement, on peut donc écrire

$$|u_2\rangle = \begin{pmatrix} (-1)^{f(00)} & 0 & 0 & 0 \\ 0 & (-1)^{f(01)} & 0 & 0 \\ 0 & 0 & (-1)^{f(10)} & 0 \\ 0 & 0 & 0 & (-1)^{f(11)} \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}. \tag{A.7}$$

Etape 3 : porte de Hadamard

On applique donc une porte de hadamard à $|u_2\rangle$:

$$|u_3\rangle = \frac{1}{2\sqrt{2}} H \left((-1)^{f(00)} |00\rangle + (-1)^{f(01)} |01\rangle + (-1)^{f(10)} |10\rangle + (-1)^{f(11)} |11\rangle\right). \tag{A.8}$$

Nous sommes sur une porte de hadamard pour 2 qubits, ce qui donne la relation matricielle suivante pour $|u_3\rangle$:

$$
\begin{aligned}
|u_3\rangle &= \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} (-1)^{f(00)} \\ (-1)^{f(01)} \\ (-1)^{f(10)} \\ (-1)^{f(11)} \end{bmatrix}, \\
&= \frac{1}{4}\begin{bmatrix} (-1)^{f(00)} + (-1)^{f(01)} + (-1)^{f(10)} + (-1)^{f(11)} \\ (-1)^{f(00)} - (-1)^{f(01)} + (-1)^{f(10)} - (-1)^{f(11)} \\ (-1)^{f(00)} + (-1)^{f(01)} - (-1)^{f(10)} - (-1)^{f(11)} \\ (-1)^{f(00)} - (-1)^{f(01)} - (-1)^{f(10)} + (-1)^{f(11)} \end{bmatrix}. \qquad \text{(A.9)}
\end{aligned}
$$

Si f est constante , alors $(-1)^{f(00)} = (-1)^{f(01)} = (-1)^{f(10)} = (-1)^{f(11)}$. En fonction du fait que $f = 0$ ou bien $f = 1$:

$$
|u_3\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{ ou bien } |u_3\rangle = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \qquad \text{(A.10)}
$$

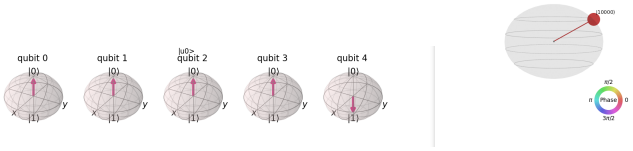
On a donc une probabilité de 1 de mesurer l'état $|00\rangle$.

En revanche, si f est équilibrée, la moitié des valeurs vont valoir $(-1)^0 = 1$ et l'autre moitié $(-1)^1 = -1$. La première ligne du vecteur $|u_3\rangle$ donne donc systématiquement 0, on ne mesure donc jamais l'état $|00\rangle$.

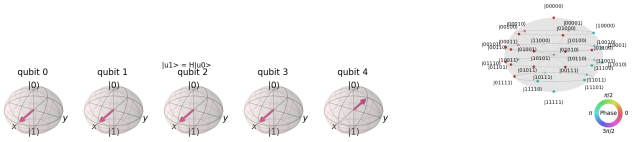
A.2 Visualisation géométrique

Reprenons cet algorithme avec $n = 4$ qubits et affichons l'évolution des états des qubits avec des sphères de bloch.

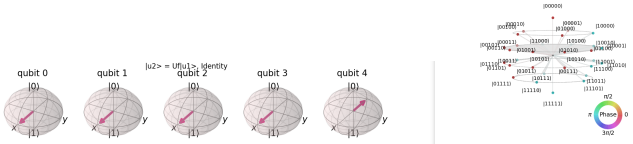
A.2.1 Fonction constante $f_0(x) = 0$



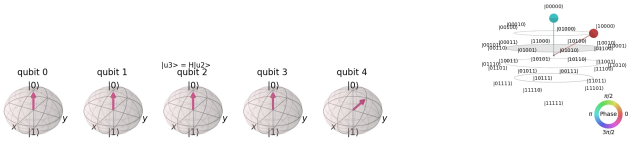
(a) $|u0\rangle = |00001\rangle$



(b) $|u1\rangle = H\,|u0\rangle = |++++-\rangle$



(c) $|u2\rangle = U_f\,|u1\rangle = |++++-\rangle$



(d) $|u3\rangle = H\,|u2\rangle = |0000\rangle\,|-\rangle$

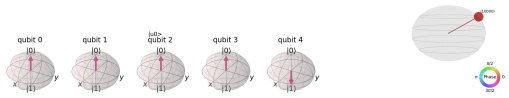
FIGURE A.1 – Evolution des états pour une fonction f constante, vecteurs d'états séparés et réunis

On voit ici l'ensemble des états que prends le registre de sortie. Dans le cas constant, on se retrouve bien à mesure exclusivement la valeur 0 (pour

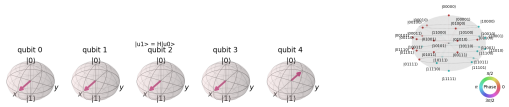
rappel, on ne mesure pas le dernier qubit qui est constant à 1). Pour les deux étapes intermédiaires, on visualise bien qu'on se retrouve dans une certaine superposition des états possibles. A noter, la visualisation présentée dans la colonne de gauche montre les états séparés. On peut rappeler que cette représentation n'est possible que parce que le 4-qubit est dans un état séparable. La représentation serait impossible si il était dans un état superposé (par exemple, si $|u\rangle = \frac{1}{2}(|10001\rangle + |01001\rangle)$).

A.2.2 Fonction équilibrée quelconque $f_1(x)$

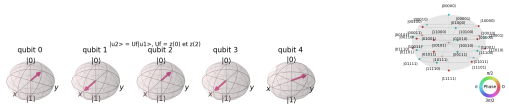
La figure suivante présente la même visualisation, pour une fonction équilibrée :



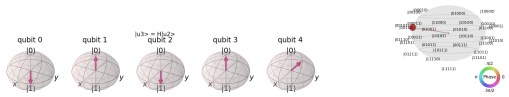
(a) $|u0\rangle = |0001\rangle$



(b) $|u1\rangle = H\,|u0\rangle = |++++-\rangle$



(c) $|u2\rangle = U_f\,|u1\rangle = |-+-+-\rangle$



(d) $|u3\rangle = H\,|u2\rangle = |1010\rangle\,|-\rangle$

FIGURE A.2 – Evolution des états pour une fonction f équilibrée, vecteurs d'états séparés

On voit bien sur cette figure que dans le cas d'une fonction équilibrée, l'état va se situer à un des points indiqués sur la sphère de Bloch mais jamais sur le point 0.

Annexe B

Algorithme dérivé de Deutsch-Jozsa : Bernstein-Vazirani

B.1 Problème à résoudre

Soient x et s tels que $x, s \in \{0,1\}^n$.
On pose une fonction f définie par :

$$\begin{array}{lccl} f: & x & \to & y = s \cdot x \pmod 2 = x_1 s_1 + x_2 s_2 + \cdots + x_n s_n \\ f: & \{0,1\}^n & \to & \{0,1\}, \end{array}$$

Exemple 2. *Soit s le mot booléen suivant : $s = 10$. La fonction f a donc la table de véritée suivante :*

(x_1, x_2)	s	$f(x_1, x_2)$
$(0,0)$	10	0
$(0,1)$	10	0
$(1,0)$	10	1
$(1,1)$	10	1

On observe que le résultat est de 1 pour les entrées (x_1, x_2) où l'emplacement des 1 correspond à ceux de s.

Problème 4 (Bernstein-Vazirani)**.** *Etant donné un mot s secret, et la fonction f implémentant l'opération décrite précédemment, comment peut on retrouver s en le moins d'évaluations de f possibles ?*

B.1.1 Solution classique

Dans le cas classique, on va devoir évaluer au pire toutes les valeurs possibles de s pour trouver sa valeur, soit n évaluations de f. C'est un algorithme de complexité $\mathcal{O}(n)$.

B.1.2 Solution quantique

Dans le cas quantique, ce problème se résout en une seule évaluation quantique de f. L'algorithme reprends celui de Deutsch-Jozsa en changeant la fonction appliquée dans l'oracle quantique.

Initialisation

On commence avec : $|u_0\rangle = (|0\rangle^{\bigotimes n})$: n-qubits à $|0\rangle$.

Etape 1

On applique une porte de Hadamard à $|u_0\rangle$ pour avoir un état équiprobable : $|u_1\rangle = H\,|u_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$.

Etape 2

On applique l'oracle quantique suivant à $|u_1\rangle$:

$$o : |x\rangle\,|y\rangle \mapsto |x\rangle\,|y \oplus (s \cdot x \pmod 2)\rangle\,.$$

En suivant exactement le même raisonnement que pour Deutsch-Jozsa, on arrive à l'expression suivante :

$$|u_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{s\cdot x \pmod 2}\,|x\rangle\,. \tag{B.1}$$

Etape 3

De la même façon à Deutsch-Jozsa, on applique une porte Hadamard à chaque qubit sortant, ce qui donne :

$$
\begin{aligned}
|u_3\rangle &= \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{s\cdot x \pmod 2} \left(\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} (-1)^{x.y} |y\rangle \right), \\
&= \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{s\cdot x \pmod 2} \left(\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} (-1)^{x.y} |y\rangle \right), \\
&= \frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{y=0}^{2^n-1} (-1)^{(s\cdot x \pmod 2)+x.y} |y\rangle . \qquad \text{(B.2)}
\end{aligned}
$$

Et on peut prouver que $\frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{y=0}^{2^n-1} (-1)^{(s\cdot x \pmod 2)+x.y} |y\rangle$ est égal à $|s\rangle$.

B.1.3 Exemple

Prenons par exemple $s = (10)_2 = 2_{10}$, soit $f(x) = 2 \cdot x \pmod 2$.

Etape 1 : porte de Hadamard

On commence avec $|u_0\rangle = |00\rangle$. La première étape est l'application de la porte d'hadamard à $|u_0\rangle$:

$$
\begin{aligned}
|u_1\rangle = H|u_0\rangle &= H|0\rangle \otimes H|0\rangle , \\
&= \frac{1}{2} ((|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle)) , \\
&= \frac{1}{2} \{|00\rangle + |01\rangle + |10\rangle + |11\rangle\} \qquad \text{(B.3)}
\end{aligned}
$$

Etape 2 : oracle quantique

On applique à $|u_1\rangle$ l'oracle quantique $|x\rangle |y\rangle \to |x\rangle |y \oplus (s \cdot x \pmod 2)\rangle =:$

$$
\begin{aligned}
|u_2\rangle &= \frac{1}{2}((-1)^{10\cdot 00 \pmod 2} |00\rangle + (-1)^{10\cdot 01 \pmod 2} |01\rangle + (-1)^{10\cdot 10 \pmod 2} |10\rangle + (-1)^{10\cdot 11 \pmod 2} |11\rangle \\
&= \frac{1}{2}((-1)^0 |00\rangle + (-1)^0 |01\rangle + (-1)^1 |10\rangle + (-1)^1 |11\rangle), \\
&= \frac{1}{2}(|00\rangle + |01\rangle - |10\rangle - |11\rangle) \qquad \text{(B.4)}
\end{aligned}
$$

Etape 3 : porte de Hadamard

On applique donc une porte de hadamard à $|u_2\rangle$:

$$|u_3\rangle = \frac{1}{2} H \left((|00\rangle + |01\rangle - |10\rangle + |11\rangle) \right). \tag{B.5}$$

Nous sommes sur une porte de hadamard pour 2 qubits, ce qui donne la relation matricielle suivante pour $|u_3\rangle$:

$$\begin{aligned} |u_3\rangle &= \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}, \\ &= \frac{1}{4} \begin{bmatrix} 0 \\ 0 \\ 4 \\ 0 \end{bmatrix}. \end{aligned} \tag{B.6}$$

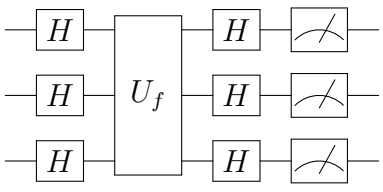
Lors de la mesure, on va obtenir l'état $|10\rangle$ avec une probabilité de 1, qui était bien notre mot binaire s de départ.

On peut observer que, lors de l'application de la porte de Hadamard à $|u_2\rangle$, on obtient la superposition d'état suivante : $|00\rangle + |01\rangle - |10\rangle - |11\rangle$. Cela correspond à la troisième ligne de la matrice de Hadamard, correspondant au $|s\rangle$ voulu. Dans tout les cas, peut importe le s choisi, on obtiendra une superposition d'état correspondant à une des lignes de la matrice, forçant à 0 les probabilités de tout les états sauf de celui indiqué.

B.1.4 Implémentation du circuit

Circuit global

L'implémentation du circuit quantique pour cet algorithme est très similaire à celui de Deutsch-Jozsa, à la différence qu'on a un qubit de moins :



Implémentation de l'oracle

Prenons le cas où $n = 2$. La matrice correspondant à la porte U_f va avoir 4 possibilité pour obtenir, comme on l'a dit lors de l'exemple, une des 4 lignes

de la matrice de Hadamard :

$$U_{f_{00}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, U_{f_{01}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, U_{f_{10}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, U_{f_{11}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

On remarque que ces quatres matrices sont en fait des produits tensoriels de deux matrices correspondant à des portes à 1 qubit :

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

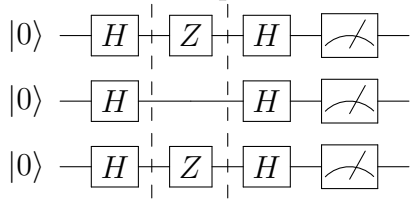
Pour $n = 2$, on a $s \in \{00, 01, 10, 11\}$. En reprenant les matrices correspondantes, on obtient les produits tensoriels suivant :

$$U_{f_{00}} = I \otimes I, U_{f_{01}} = I \otimes Z, U_{f_{10}} = Z \otimes I, U_{f_{11}} = Z \otimes Z.$$

On peut généraliser sur l'implémentation en disant :

$$U_f = \bigotimes_{i=0}^{n} U_i,\ U_i = \begin{cases} I & \text{si } s_i = 0 \\ Z & \text{si } s_i = 1 \end{cases}. \tag{B.7}$$

Un exemple d'implémentation complète serait alors (pour $s = 101$) :



Annexe C

Algorithme de Grover

C.1 Rappels d'algèbre : projection et reflection

Soient deux vecteurs $\overrightarrow{u}$ et $\overrightarrow{v}$, avec $\overrightarrow{v}$ normalisé.

Définition 3. *La matrice de projection P de $\overrightarrow{u}$ sur $\overrightarrow{v}$ est définie par $P = \overrightarrow{v} \cdot \overrightarrow{v^T}$.*

Définition 4. *La matrice de reflection R de $\overrightarrow{u}$ par rapport à $\overrightarrow{v}$ est définie par $R = 2\overrightarrow{v} \cdot \overrightarrow{v^T} - I$.*

Exemple 3. *Prenons* $\overrightarrow{u} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ *et* $\overrightarrow{v} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$.

On projete $\overrightarrow{u}$ sur $\overrightarrow{v}$:

$P = \frac{\overrightarrow{v} \cdot \overrightarrow{v^T}}{\|v\|^2} = \begin{bmatrix} \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} \\ \frac{-2}{\sqrt{5}} & \frac{4}{\sqrt{5}} \end{bmatrix}$

Soit : $\overrightarrow{u_v} = P\overrightarrow{u} = \begin{bmatrix} -0.8 \\ 1.6 \end{bmatrix}$

Exemple 4. *Prenons à nouveau* $\overrightarrow{u} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ *et* $\overrightarrow{v} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$. *On effectue une reflection de $\overrightarrow{u}$*

$R = 2 \times \frac{\overrightarrow{v} \cdot \overrightarrow{v^T}}{\|v\|^2} - I = 2 \times \begin{bmatrix} \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} \\ \frac{-2}{\sqrt{5}} & \frac{4}{\sqrt{5}} \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

La première étape est la double projection $2 \times P$, ce qui donne le vecteur $\begin{bmatrix} -1.6 \\ 3.2 \end{bmatrix}$.

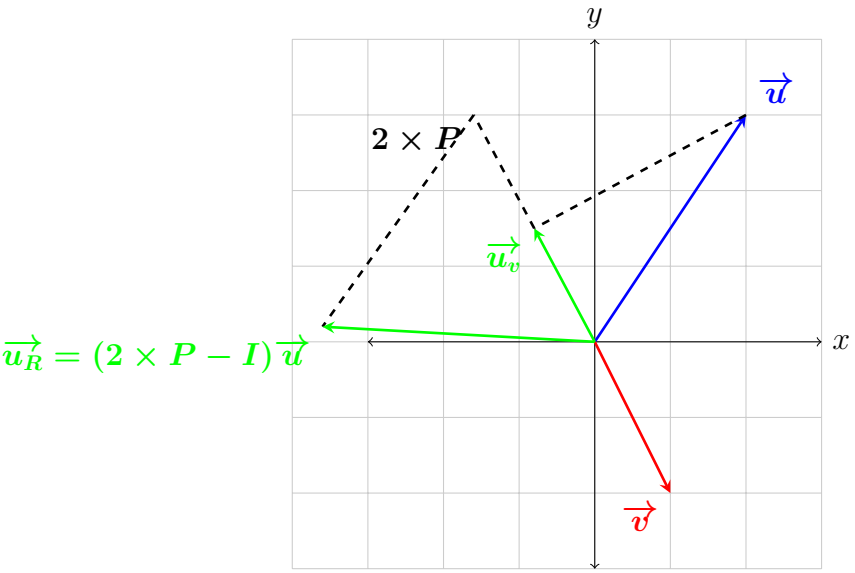
La deuxième étape est d'enlever le vecteur initial, ce qui donne le vecteur $\overrightarrow{u_R} = \begin{bmatrix} -3.6 \\ 0.2 \end{bmatrix}$.

On peut vérifier les angles θ_{UV} et θ_{VU_R} :

$\theta_{UV} = \arccos(\frac{u \cdot v}{\|u\|\|v\|}) = \arccos(\frac{-4}{\sqrt{13} \times \sqrt{5}}) = 119.7°$

$\theta_{VU_R} = \arccos(\frac{v \cdot u_R}{\|v\|\|u_R\|}) = \arccos(\frac{-4}{\sqrt{5} \times \sqrt{13}}) = 119.7°$

Les deux angles sont bien égaux, on a effectué une reflection.



C.2 Problème à résoudre

Soit une base de données non triée à N entrées. Nous voulons trouver un algorithme permettant de chercher efficacement un enregistrement dans cette base.

C.2.1 Principe de l'algorithme

L'algorithme de Grover permet de résoudre ce problème en quantique, en disposant de N qubits intriqués pour calculer 2^N état (donc si on a N entrées dans la base, il nous faut $log_2(N)$ qubits intriqués). Dans le cas de cet algorithme, on considère le problème suivant :

On marque $\{0, 1, 2, ..., N-1\}$ les enregistrements de la base de données, et on dénote ω l'état inconnu recherché. On dispose de la fonction suivante :

$$f(x) = \begin{cases} 1, & \text{si x vérifie le critère } \omega \\ 0, & \text{sinon} \end{cases}$$

A la fin, on obtient un set de résultat. Or, lors de la mesure on va avoir au hasard une des solutions suivant les probabilités de chaque état, alors qu'on cherche juste à savoir la (ou les) bonnes solutions. On rajoute donc une amplification d'amplitude permettant d'augmenter les probabilités des bons résultats et de diminuer celles des mauvais.

Initialisation

On commence avec : $|u_0\rangle = (|0\rangle^{\bigotimes n}) \otimes |1\rangle$: n-qubits à $|0\rangle$ et 1-qubit à $|1\rangle$

Etape 1

On applique une porte de Hadamard à $|u_0\rangle$ pour avoir un état équiprobable : $|u_1\rangle = H|u_0\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle (|0\rangle - |1\rangle)$

On pose alors $|s\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$

Etape 2 : opérateurs de Grover

On définit les deux opérateurs suivants :

$U_w = I - 2|w\rangle\langle w|$, avec w état cible correspondant à la solution du problème (amplitude de 1 sur l'état visé, amplitude nulle sur le reste)

$U_s = 2|s\rangle\langle s| - I$

Remarque 3. *On reconnait ici que ces deux opérateurs sont semblables à la reflection vue dans la partie 1.*

On effectue ici un changement de base : au lieu de continuer les calculs dans la base canonique $\{|0\rangle, |1\rangle\}$, on se place dans la base $\{|w\rangle, |s\rangle\}$

Inversion d'amplitude L'opérateur U_w effectue l'inversion de l'amplitude de l'état cible, tandis que l'opérateur U_s effectue le miroir des amplitudes par rapport à la moyenne.

On applique U_w puis U_s :

$U_w|s\rangle = (I - 2|w\rangle\langle w|)|s\rangle = |s\rangle - 2|w\rangle\langle w|s\rangle$

Or, $\langle w|s\rangle$ est un produit scalaire. $|w\rangle$ est définit plus haut, et $|s\rangle$ est l'état équiprobable obtenu après la porte de hadamard. Le résultat est donc $\langle w|s\rangle = \frac{1}{\sqrt{2^n}}$. On peut donc réécrire :

$|u_3\rangle = U_w|s\rangle = |s\rangle - \frac{2}{\sqrt{2^n}}|w\rangle$

Miroir à la moyenne On applique ensuite l'opérateur U_s au résultat de U_w. On peut voir qu'en pratique U_s effectue un miroir de $|u_3\rangle$ par rapport à $|s\rangle$.

$$\begin{aligned} U_s |u_3\rangle &= (2 |s\rangle \langle s| - I)(|s\rangle - \frac{2}{\sqrt{2^n}} |w\rangle) \\ &= 2 |s\rangle \langle s|s\rangle - |s\rangle - \frac{4}{\sqrt{2^n}} |s\rangle \langle s|w\rangle + \frac{2}{\sqrt{2^n}} |w\rangle \\ &= 2 |s\rangle - |s\rangle + \frac{4}{\sqrt{2^n}} \times \frac{1}{\sqrt{2^n}} |s\rangle + \frac{2}{\sqrt{2^n}} |w\rangle \\ &= |s\rangle - \frac{4}{2^n} |s\rangle + \frac{2}{\sqrt{2^n}} |w\rangle \\ |u_4\rangle &= \frac{2^n - 4}{2^n} |s\rangle + \frac{2}{\sqrt{2^n}} |w\rangle \end{aligned} \tag{C.1}$$

Plus généralement, cette application de U_w puis U_s reviens à appliquer la matrice suivante à l'état d'entrée, dans la base $\{|w\rangle, |s\rangle\}$: $\begin{bmatrix} 1 & \frac{2}{\sqrt{2^n}} \\ \frac{-2}{\sqrt{N}} & \frac{2^n-4}{2^n} \end{bmatrix}$

C.2.2 Exemple

Prenons par exemple une base de données de 4 bits ($n = 4$), avec l'état $|w\rangle$ cible valant l'état $|0100\rangle$ (amplitude de 1 sur cet état, et de 0 sur l'ensemble de 15 autres).

On initialise un (n+1)-qubit à l'état suivant :

$$|u_0\rangle = |00001\rangle \tag{C.2}$$

Etape 1

On applique la porte de Hadamard à l'état initial $|u_0\rangle$:

$$|u_1\rangle = \frac{1}{16} \sum_{x=0}^{15} |x\rangle (|0\rangle - |1\rangle) \tag{C.3}$$

On obtient donc les deux états formant notre base pour les calculs suivant : $|s\rangle = |u_1\rangle$ et $|w\rangle$.

Etape 2 : Opérateur de Grover

On applique la transformation $U_s U_w = \begin{bmatrix} 1 & \frac{2}{\sqrt{2^n}} \\ \frac{-2}{\sqrt{2^n}} & \frac{2^n-4}{2^n} \end{bmatrix}$ pour $n = 4$ soit $U_s U_w = \begin{bmatrix} 1 & \frac{1}{2} \\ -\frac{1}{2} & \frac{3}{4} \end{bmatrix}$:

$$|u_2\rangle = U_s U_w \cdot |s\rangle = \frac{3}{4}|s\rangle + \frac{1}{2}|w\rangle \tag{C.4}$$

On voit que l'état cible $|w\rangle$ est passé d'une amplitude de 0 à une amplitude de 0.5. On peut effectuer l'opération plusieurs fois pour obtenir un résultat voulu. La figure suivante montre l'évolution des amplitudes de $|s\rangle$ et de $|w\rangle$ pour $n = 16$, pour 1000 itérations de l'opérateur. On observe qu'on arrive à l'état voulu $|w\rangle$ mais qu'on ne reste pas à cet état une fois atteint. Cela montre bien qu'il y a un nombre optimal d'itérations à effectuer, à ne pas dépasser. (la courbe verte sert d'indicateur, pour vérifier qu'on reste dans un état valide où la somme des amplitudes vaut bien 1)

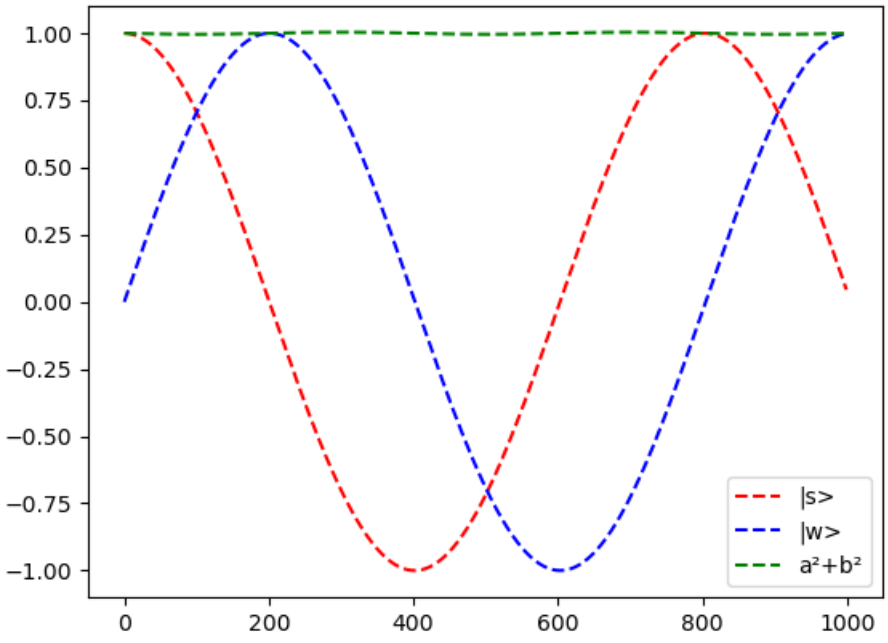


FIGURE C.1 – Evolution des amplitudes pour n=16, sur 1000 itérations

C.2.3 Implémentation

Simulation sur un ordinateur classique

Data: w vector of size 2^n of 0 with target index to 1 ;
Output: x vector of amplitudes (largest amplitude corresponding to wanted index)
begin
  s vector of size 2^n of $\frac{1}{\sqrt{2^n}}$ $N \longleftarrow 2^n$;
  $N_{iter} \longleftarrow floor(\frac{\pi}{4}\sqrt{N})$;
  `/* Compute grover operator */`
  $U_w \longleftarrow I_N - 2w \cdot w^T$;
  $U_s \longleftarrow 2s \cdot s^T - I_N$;
  $g \longleftarrow U_s \cdot U_w$;
  $x \longleftarrow s$;
  `/* Apply grover operator` N_{iter} `times */`
  for $i = 0$ **to** N_{iter} **do**
    $x \longleftarrow g \cdot x$;
  end
end

Algorithme quantique

```
import numpy as np
import math
import random
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit

var_qubits = QuantumRegister(4, name='v')
clause_qubits = QuantumRegister(4, name='c')
output_qubits = QuantumRegister(1, name='out')
cbits = ClassicalRegister(4, name='cbits')
qc = QuantumCircuit(var_qubits, clause_qubits, output_qubits, cbits)
qc.initialize([1, -1]/np.sqrt(2), output_qubits)
qc.h(var_qubits)
qc.barrier()
```

Annexe D

Conclusion

Bibliographie

[1] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation," *Proceedings of the Royal Society of London A*, vol. 439, pp. 553–558, 1992.

[2] E. Bernstein and U. Vazirani, "Quantum complexity theory," *SIAM Journal on Computing*, vol. 26, pp. 1411–1473, 1997.

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," p. 212–219, 1996.

[4] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, pp. 1484–1509, 1997.