Chapitre 3 : Algorithme quantique d'estimation de phase

Informatique quantique pour la recherche opérationnelle
Dylan Laplace Mermoud - ensIIE

2025

Brèves notes de cours adaptées du chapitre 5 du livre "Quantum Computation and Quantum Information" de Michael A. Nielsen et Isaac L. Chuang.

1 Introduction

L'un des algorithmes les plus importants du calcul classique est la transformée de Fourier rapide. Elle calcule la transformée de Fourier discrète d'une liste de n éléments avec une complexité de $O(n \log n)$ et est massivement utilisée dans les domaines des télécommunications et du traitement du signal, ainsi que dans les techniques de compression numérique ayant mené au format jpeg.

En 1965, Cooley et Tukey publient leur article qui popularisera cet algorithme, bien que cette méthode de calcul de la transformée de Fourier discrète soit connue depuis au moins 1805 grâce à Gauss. En 1994, Gilbert Strang a décrit la transformée de Fourier rapide comme étant l'algorithme le plus important de notre époque, et est inclus dans la liste des 10 algorithmes les plus importants du 20e siècle éditée par le magazine IEEE *Computing in Science and Engineering*, aux côtés de l'algorithme du simplexe ou de la décomposition QR par exemple.

Il n'est donc pas surprenant de voir qu'un algorithme quantique permettant de calculer la transformée de Fourier discrète soit l'ingrédient principal du plus important algorithme quantique à ce jour, l'algorithme de Shor. Ce dernier permet de factoriser un entier naturel n en temps $O((\log n)^3)$ et en espace $O(\log n)$. De nombreux cryptosystèmes à clé publique reposent sur le fait qu'il est difficile pour un ordinateur classique de factoriser un grand entier, l'algorithme classique le plus efficace à ce jour est celui du crible du corps de nombre généralisé, a une complexité sous-exponentielle par rapport à la taille de l'entier à factoriser.

2 Estimation de phase

La partie quantique de l'algorithme de Shor repose sur l'algorithme quantique d'estimation de phase qui permet, étant donné un opérateur unitaire U et un de ses vecteurs propres $|\varphi\rangle$, de calculer une approximation de la valeur propre associée à $|\varphi\rangle$. L'opérateur U étant unitaire, ses valeurs propres ont pour module 1, ainsi la valeur propre associée à $|\varphi\rangle$ peut s'écrire $\exp(2i\pi\phi)$, et $\phi \in [0, 1)$ est la phase que l'on souhaite connaitre. Autrement dit, l'algorithme d'estimation de phase nous donne ϕ qui satisfait l'équation suivante

$$U|\varphi\rangle = \exp(2i\pi\phi)|\varphi\rangle, \qquad \text{avec} \quad 0 \leq \phi < 1.$$

L'algorithme d'estimation de phase est également utilisé dans un autre important algorithme quantique, l'algorithme HHL (pour Harrow-Hassidim-Lloyd, les trois auteurs), qui permet de calculer la solution d'un système d'équations linéaires en tant qu'état.

Pour utiliser l'algorithme quantique d'estimation de phase, on suppose que

- ▷ l'on dispose d'un registre dont l'état est précisément le vecteur propre $|\varphi\rangle$,
- ▷ l'on puisse appliquer des opérations controlées de l'opérateur unitaire U ainsi que de ces puissances U^{2^j}, pour j un entier positif.

L'algorithme d'estimation de phase utilise deux registres. Le premier registre contient q qubits, dont l'état initial est $|\underline{0}\rangle = |0\rangle^{\otimes q}$. Le nombre q dépend de deux choses : la précision souhaitée pour l'approximation de la phase, et la probabilité de succès souhaitée de l'algorithme. Le second registre est initialisé à $|\varphi\rangle$, et contient autant de qubits que nécéssaire pour cela.

Schématiquement, l'algorithme nous donne une application linéaire transformant

$$|\underline{0}\rangle|\varphi\rangle \mapsto |\phi\rangle|\varphi\rangle$$

Notons qu'il est également possible de remplacer le vecteur propre sur le second registre par une combinaison linéaire de vecteurs propres. En effet, comme l'opérateur unitaire U est linéaire, il s'applique sur chaque vecteur de la combinaison linéaire. On se retrouve alors avec une superposition des états encodant les phases à mesurer :

$$|\underline{0}\rangle\left(\sum_k \gamma_k|\varphi_k\rangle\right) = \sum_k \gamma_k|\underline{0}\rangle|\varphi_k\rangle \mapsto \sum_k |\phi_k\rangle|\varphi_k\rangle$$

Lorsqu'on mesure ce registre, on obtient l'une des phases aléatoirement, et la probabilité de mesurer une phase est

donnée par le carré du module de son coefficient dans la combinaison linéaire, i.e., $|\gamma_k|^2$. Cela sera très utile lorsque nous étudierons les applications de l'algorithme d'estimation de phase.

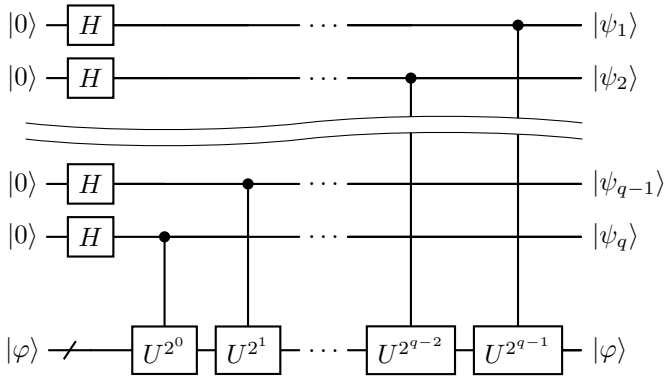

Figure 1: La première étape de l'algorithme d'estimation de phase, avec $|\psi_j\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle + \exp\left(2\pi i 2^{q-j}\phi\right)|1\rangle\right]$

L'estimation de phase s'effectue en deux temps. Premièrement, on applique le circuit décrit par la figure 1. Le circuit commence avec une couche de portes Hadamard sur le premier registre pour créer une superposition uniforme de tous les états de la base computationelle, suivi de l'application des opérations controlées de U, avec U élevée à la puissance 2^{j-1} à l'étape j.

L'état final après cette première étape du premier registre est donné par

$$\begin{aligned}&\frac{1}{2^{q/2}}\bigotimes_{m=1}^{q}\left(|0\rangle + \exp\left(2i\pi 2^{q-m}\phi\right)|1\rangle\right)\\ &= \frac{1}{2^{q/2}}\sum_{k=0}^{2^q-1}\exp\left(2i\pi k\phi\right)|k\rangle\,,\end{aligned}$$

tandis que l'état du second registre n'a pas changé, car il est un vecteur propre de U, et donc de chacune des puissances U^{2^j} pour j un entier positif.

Exemple 1. Regardons de plus près ce qui arrive au sous-système formé par le $(j+1)$-ième qubit du premier registre et le second registre lorsque nous appliquons la porte controlée correspondantes. On démarre avec l'état

$$|0\rangle|\varphi\rangle\,,$$

puis, après application de la porte Hadamard, on obtient

$$\frac{1}{\sqrt{2}}\left(|0\rangle|\varphi\rangle + |1\rangle|\varphi\rangle\right).$$

L'application de l'opérateur U^{2^j} étant conditionné par le fait que l'état du qubit de contrôle soit égal à $|1\rangle$, on obtient, après son application, l'état

$$\frac{1}{\sqrt{2}}\left(|0\rangle|\varphi\rangle + |1\rangle U^{2^j}|\varphi\rangle\right).$$

Comme $|\varphi\rangle$ est un vecteur propre de U, il est également un vecteur propre de U^{2^j}. La valeur propre de U^{2^j} correspondante à $|\varphi\rangle$ s'obtient récursivement :

$$\begin{aligned}U^k|\varphi\rangle &= U^{k-1}U|\varphi\rangle\\ &= U^{k-1}\exp(2i\pi\phi)|\varphi\rangle\\ &= \exp(2i\pi\phi)U^{k-1}|\varphi\rangle\\ &= \left(\exp(2i\pi\phi)\right)^k|\varphi\rangle\,.\end{aligned}$$

Ainsi, la valeur propre de U^{2^j} correspondante à $|\varphi\rangle$ est

$$\left(\exp\left(2i\pi\phi\right)\right)^{2^j} = \exp\left(2i\pi 2^j\phi\right).$$

On peut alors réécrire l'état de notre sous-système comme

$$\begin{aligned}&\frac{1}{\sqrt{2}}\left(|0\rangle|\varphi\rangle + |1\rangle U^{2^j}|\varphi\rangle\right)\\ &= \frac{1}{\sqrt{2}}\Big(|0\rangle|\varphi\rangle + |1\rangle\exp(2i\pi 2^j\phi)|\varphi\rangle\Big).\end{aligned}$$

Par linéarité, on peut écrire cet état :

$$|\psi_j\rangle|\varphi\rangle \quad \text{avec} \quad |\psi_j\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + \exp(2i\pi 2^j\phi)|1\rangle\right),$$

qui est précisément l'état décrit sur la figure 1. ◊

La deuxième étape de l'algorithme d'estimation de phase est l'application de l'inverse de la transformée de Fourier quantique, sur le premier registre uniquement. La troisième et dernière étape de cet algorithme est simplement la lecture de l'état du premier registre en mesurant dans la base computationelle.

3 Transformée de Fourier quantique

Dans le contexte de ce cours, la transformée de Fourier quantique est l'implémentation de la transformée de Fourier discrète d'une liste de n nombres complexes stockés comme amplitudes d'un état quantique. Classiquement, la transformée de Fourier discrète d'un vecteur $x = (x_0, \ldots, x_{n-1})$ est un vecteur de même taille $y = (y_0, \ldots, y_{n-1})$ défini par

$$y_k \coloneqq \frac{1}{\sqrt{n}}\sum_{j=0}^{n-1}\omega_n^{jk}x_j,$$

avec $\omega_n = e^{2i\pi/n}$. La transformée de Fourier quantique est alors l'algorithme quantique qui envoie l'état $|j\rangle$ de la base computationelle sur l'état

$$|j\rangle \longmapsto \frac{1}{\sqrt{n}}\sum_{k=0}^{n-1}\omega_n^{jk}|k\rangle\,,$$

ce qui nous permet d'écrire la transformée de Fourier comme étant l'opérateur F_n défini par

$$F_n \coloneqq \frac{1}{\sqrt{n}} \sum_{j,k=0}^{n-1} \omega_n^{jk} \left|k\right\rangle\!\left\langle j\right|,$$

où $\langle j|$ est la forme linéaire agissant sur l'espace des états de la manière suivante :

$$\langle j| \left(|k\rangle\right) = \langle j|k\rangle = \begin{cases} 1, & \text{si } j = k, \\ 0, & \text{sinon.} \end{cases} \tag{1}$$

Ainsi, les $\{\langle j| \mid 0 \leq j \leq n-1\}$ forment une base orthonormée de l'espace dual de celui des états dont la base est $\{|k\rangle \mid 0 \leq k \leq n-1\}$. On peut alors voir l'élément $|k\rangle\langle j|$ comme une matrice M où toutes les entrées sont 0 sauf M_{kj} qui vaut 1.

Proposition 3.1. *La transformée de Fourier F_n est unitaire.*

Proof. On va prouver que $F_n^\dagger F_n = \mathbf{1}$ (le cas $F_n F_n^\dagger = \mathbf{1}$ est quasiment identique). Premièrement, remarquons que

$$\overline{\omega_n^{jk}} = \exp(-2i\pi jk/n) = \omega_n^{-jk}, \quad \text{et} \quad \omega_n^{kj} = \omega_n^{jk},$$

ce qui implique que

$$F_n^\dagger = \frac{1}{\sqrt{n}} \sum_{j,k=0}^{n-1} \omega_n^{-jk} \left|k\right\rangle\!\left\langle j\right|.$$

On a alors

$$F_n^\dagger F_n = \frac{1}{n} \sum_{j',k'=0}^{n-1} \sum_{j,k=0}^{n-1} \omega_n^{j'k'-jk} \left|j\right\rangle\!\left\langle k|k'\right\rangle\!\left\langle j'\right|.$$

En utilisant l'équation (1), on voit que seuls les termes où $k' = k$ sont non nuls. Le produit matriciel s'écrit alors

$$F_n^\dagger F_n = \frac{1}{n} \sum_{j,j'=0}^{n-1} \sum_{k=0}^{n-1} \omega_n^{(j'-j)k} \left|j\right\rangle\!\left\langle j'\right|$$

Supposons, temporairement, que

$$\frac{1}{n} \sum_{k=0}^{n-1} \omega_n^{(j'-j)k} = \begin{cases} 1, & \text{si } j = j', \\ 0, & \text{sinon.} \end{cases} \tag{2}$$

Le produit se simplifie alors comme suit :

$$F_n^\dagger F_n = \sum_{j=0}^{n-1} \left|j\right\rangle\!\left\langle j\right|,$$

ce qui est en effet égal à la matrice identité $\mathbf{1}$. Démontrons maintenant l'équation (2). Rappelons que les séries géométriques tronquées ont comme formule

$$\sum_{k=0}^{p-1} \alpha^k = \begin{cases} \frac{1-\alpha^p}{1-\alpha}, & \text{si } \alpha \neq 1, \\ p, & \text{si } \alpha = 1. \end{cases}$$

Comme j et j' sont plus petits que n, on a que $\omega_n^{(j'-j)}$ n'est égal à 1 que lorsque $j' = j$. Si $j' \neq j$, alors,

$$\sum_{k=0}^{n-1} \omega_n^{(j'-j)k} = \frac{1-\omega_n^{(j'-j)n}}{1-\omega_n^{(j'-j)}} = \frac{1-1^{(j'-j)}}{1-\omega_n^{(j'-j)}} = 0.$$

En revanche, quand $j' = j$, on a

$$\sum_{k=0}^{n-1} \omega_n^{(j'-j)k} = \sum_{k=0}^{n-1} 1^k = n.$$

En combinant les deux cas, et en divisant par n, on retombe bien sur la formule de l'équation (2). □

À partir de maintenant, par simplicité, on va supposer que $n = 2^q$, pour un entier positif q. Cela signifie que q qubits sont suffisants pour encoder n amplitudes.

Il est possible de voir l'image d'un élément de la base computationelle $|j\rangle$ à travers la transformée de Fourier comme un état produit, c'est-à-dire que l'on peut exprimer comme un produit tensoriel. Cela nous permet de faire le lien plus facilement avec l'état préparé par la première étape de l'algorithme d'estimation de phase.

Proposition 3.2. *La transformée de Fourier de l'état de la base canonique $|j\rangle$ peut s'écrire*

$$|j\rangle \mapsto \frac{1}{\sqrt{n}} \bigotimes_{\ell=1}^{q} \left(|0\rangle + \omega_{2^\ell}^j |1\rangle\right) \tag{3}$$

Proof. Commençons d'abord par décomposer la somme sur chacun des états de la base computationnelle pour une série de sommes sur chacun des qubits :

$$\begin{aligned} |j\rangle &\mapsto \frac{1}{2^{q/2}} \sum_{k=0}^{2^q-1} \omega_{2^q}^{jk} |k\rangle \\ &= \frac{1}{\sqrt{n}} \sum_{k_1=0}^{1} \cdots \sum_{k_q=0}^{1} \exp\left(2i\pi j \sum_{\ell=1}^{q} k_\ell 2^{-\ell}\right) |k_1\rangle \ldots |k_q\rangle. \end{aligned}$$

On peut réécrire l'exponentielle de cette somme de scalaires en un produit d'exponentielles :

$$\exp\left(2i\pi j \sum_{\ell=1}^{q} k_\ell 2^{-\ell}\right) = \prod_{\ell=1}^{q} \exp(2i\pi j k_\ell 2^{-\ell}) = \prod_{\ell=1}^{q} \omega_{2^\ell}^{jk_\ell}.$$

Utilisant la linéarité du produit tensoriel, on obtient :

$$\begin{aligned} |j\rangle &\mapsto \frac{1}{\sqrt{n}} \sum_{k_1=0}^{1} \cdots \sum_{k_q=0}^{1} \bigotimes_{\ell=1}^{q} \omega_{2^\ell}^{jk_\ell} |k_\ell\rangle \\ &= \frac{1}{\sqrt{n}} \bigotimes_{\ell=1}^{q} \left(\sum_{k_\ell=0}^{1} \omega_{2^\ell}^{jk_\ell} |k_\ell\rangle\right) \end{aligned}$$

Chacune de ses sommes ne contient que deux termes, on peut alors simplifier la formule précédente en écrivant :

$$|j\rangle \mapsto \frac{1}{\sqrt{n}} \bigotimes_{\ell=1}^{q} \left(|0\rangle + \omega_{2^\ell}^j |1\rangle\right)$$

ce qui donne la formule souhaitée. □

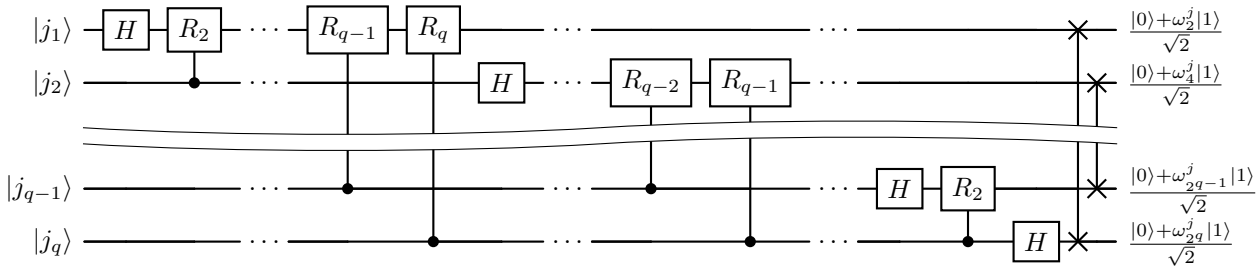

Figure 2: La transformée de Fourier quantique : circuit efficace pour calculer la transformée de Fourier discrète.

4 Algorithme complet

Rappelons qu'à la fin de la première étape de l'algorithme d'estimation de phase, nous avons le premier registre dans l'état :

$$\frac{1}{\sqrt{n}} \bigotimes_{\ell=1}^{q} \Big(|0\rangle + \exp(2i\pi 2^{q-\ell}\phi)|1\rangle \Big),$$

avec $\phi \in [0,1)$. Supposons, par simplicité, que ϕ puisse être décrit exactement avec q qubits, c'est-à-dire,

$$\phi = \frac{\phi_1}{2} + \frac{\phi_2}{4} + \ldots + \frac{\phi_q}{2^q}.$$

Les amplitudes de chacun des qubits deviennent

$$\begin{aligned}\exp(2i\pi 2^{q-\ell}\phi) &= \exp\left(2i\pi 2^{q-\ell} \sum_{k=1}^{q} \frac{\phi_k}{2^k}\right) \\ &= \prod_{k=1}^{q} \exp(2i\pi 2^{q-\ell-k}\phi_k).\end{aligned}$$

Chaque facteur de ce produit tel que $q-\ell-k \geq 0$ vaut 1, on a alors la simplification :

$$\exp\left(2i\pi 2^{q-\ell}\right) = \prod_{k=q-\ell+1}^{q} \omega_{2^{k+\ell-q}}^{\phi_k} = \prod_{m=1}^{\ell} \omega_{2^m}^{\phi_{q-\ell+m}},$$

ce qui nous donne, après utilisation de l'équation (5),

$$\exp(2i\pi 2^{q-\ell}) = \omega_{2^\ell}^{\phi}$$

Alors on peut réécrire l'état précédent :

$$\frac{1}{\sqrt{n}} \bigotimes_{j=1}^{q} \Big(|0\rangle + \omega_{2^\ell}^{\phi} |1\rangle \Big),$$

ce qui est très précisément la transformée de Fourier de $|\phi\rangle$ telle qu'écrite à l'équation (3). Ainsi, appliquer l'inverse de la transformée de Fourier au premier registre après la première étape de l'algorithme d'estimation de phase encode directement la phase (ou, plutôt, une approximation de celle-ci utilisant q qubits) écrit en binaire.

En effet, l'algorithme d'estimation de phase fonctionne même lorsque la phase $\phi \in [0,1)$ recherchée ne peut pas être écrite exactement avec q qubits. On aura alors simplement un nombre légèrement plus petit, qui peut être écrit en utilisant q qubits, et dont la différence avec ϕ est plus petite que 2^{-q}.

5 Construction de la transformée de Fourier quantique

On cherche à présent à construire le circuit quantique nous permettant d'appliquer la transformée de Fourier à l'état d'un registre de qubits, décrit par la figure 2. Pour cela, regardons de plus près comment les amplitudes $\omega_{2^l}^j$ encode le nombre j. Premièrement, rappelons que, si l'on écrit j en base 2 comme $j_1\ j_2\ \ldots\ j_q$, alors

$$j = j_1 2^{q-1} + j_2 2^{q-2} + \ldots + j_q 2^0.$$

Utilisons cette décomposition pour réécrire l'amplitude :

$$\omega_{2^\ell}^j = \omega_{2^\ell}^{j_1 2^{q-1} + j_2 2^{q-2} + \ldots + j_q} = \omega_{2^\ell}^{j_1 2^{q-1}} \omega_{2^\ell}^{j_2 2^{q-2}} \ldots \omega_{2^\ell}^{j_q} \quad (4)$$

Remarquons que pour chaque terme où $q-k \geq \ell$, on a

$$\omega_{2^\ell}^{j_k 2^{q-k}} = \exp\left(2i\pi \frac{j_k 2^{q-k}}{2^\ell}\right) = \exp\Big(2i\pi j_k 2^{q-k-\ell}\Big).$$

Comme $j_k 2^{q-k-\ell}$ est un entier, on peut écrire

$$\omega_{2^\ell}^{j_k 2^{q-k}} = \exp(2i\pi)^{j_k 2^{q-k-\ell}} = 1^{j_k 2^{q-k-\ell}} = 1.$$

Alors, dans la formule de l'équation (4), nous avons que les ℓ premiers facteurs ne contribuent pas au produit. On peut alors réécrire l'amplitude :

$$\omega_{2^\ell}^j = \omega_{2^\ell}^{j_{q-\ell+1} 2^{\ell-1}} \omega_{2^\ell}^{j_{q-\ell+2} 2^{\ell-2}} \ldots \omega_{2^\ell}^{j_q}.$$

En simplifiant les puissances de 2 en exposant et en indice, on obtient la formule suivante :

$$\omega_{2^l}^j = \omega_2^{j_{q-l+1}} \omega_4^{j_{q-l+2}} \ldots \omega_{2^l}^{j_q} = \prod_{m=1}^{l} \omega_{2^m}^{j_{q-l+m}}, \quad (5)$$

qui va être très utile par la suite. En particulier, pour $\ell = q$, on a que

$$\omega_{2^q}^j = \omega_2^{j_1} \omega_4^{j_2} \ldots \omega_n^{j_q}. \quad (6)$$

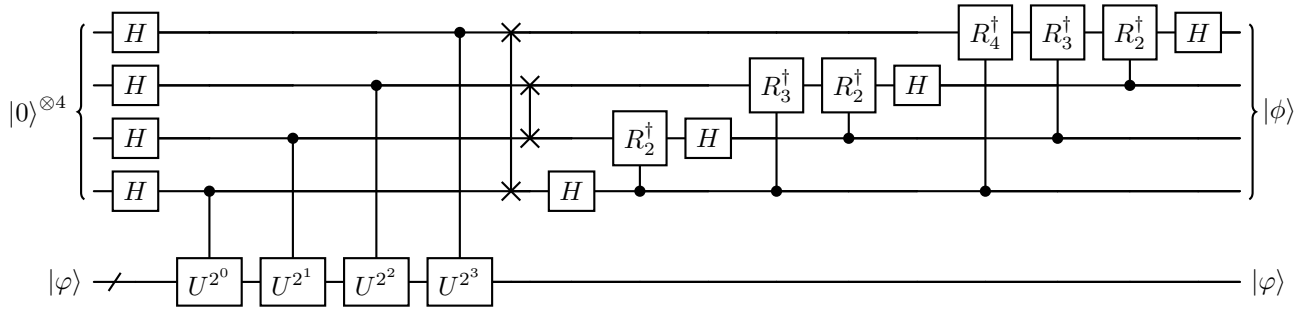

Figure 3: L'algorithme quantique d'estimation de phase avec 4 qubits de précision.

Revenons maintenant à la construction de la transformée de Fourier quantique. La proposition 3.2 nous permet de trouver une décomposition de F_n en tant que circuit quantique, décrit par la figure 2. Notons par R_k l'opérateur unitaire

$$R_k \coloneqq \begin{pmatrix} 1 & 0 \\ 0 & \omega_{2^k} \end{pmatrix},$$

qui est la porte de phase évaluée à l'angle $\theta = 2\pi/2^k$.

Pour comprendre ce qu'il se produit lorsque l'on utilise le circuit de la transformée de Fourier quantique, considérons son action sur l'état

$$|j\rangle = |j_1\rangle \dots |j_q\rangle .$$

Appliquer la porte Hadamard sur le premier qubit produit l'état

$$\frac{1}{\sqrt{2}}\Big(|0\rangle + \omega_2^{j_1}|1\rangle\Big)|j_2\rangle \dots |j_q\rangle ,$$

car l'amplitude ne peut prendre que ses deux valeurs :

$$\omega_2^{j_1} = \exp(2i\pi j_1/2) = \begin{cases} e^{i\pi} = -1, & \text{quand } j_1 = 1, \\ e^0 = 1, & \text{quand } j_1 = 0. \end{cases}$$

Appliquer la porte R_2 controlée par le deuxième qubit produit l'état

$$\frac{1}{\sqrt{2}}\Big(|0\rangle + \omega_2^{j_1}\omega_4^{j_2}|1\rangle\Big)|j_2\rangle \dots |j_q\rangle$$

On continue d'appliquer les portes de phase controlées R_3, R_4 jusqu'à R_q, chacune ajoutant un terme à l'amplitude de $|1\rangle$ du premier qubit. À la fin de cette suite de portes, on obtient

$$\frac{1}{\sqrt{2}}\Big(|0\rangle + \omega_2^{j_1}\omega_4^{j_2} \dots \omega_n^{j_q}|1\rangle\Big)|j_2\rangle \dots |j_q\rangle ,$$

que l'on peut réécrire

$$\frac{1}{\sqrt{2}}\Big(|0\rangle + \omega_n^{j}|1\rangle\Big)|j_2\rangle \dots |j_q\rangle .$$

en utilisant l'équation (6). Ensuite, on fait la même chose sur le deuxième, avec une porte de phase en moins. Après l'application de la porte Hadamard, on a

$$\frac{1}{\sqrt{4}}\Big(|0\rangle + \omega_n^{j}|1\rangle\Big)\Big(|0\rangle + \omega_2^{j_2}|1\rangle\Big)|j_3\rangle \dots |j_q\rangle .$$

On applique maintenant les portes de phase controlées R_2 jusqu'à R_{q-1} pour obtenir

$$\frac{1}{\sqrt{4}}\Big(|0\rangle + \omega_n^{j}|1\rangle\Big)\Big(|0\rangle + \omega_2^{j_2}\omega_4^{j_3} \dots \omega_{2^{q-1}}^{j_q}|1\rangle\Big)|j_3\rangle \dots |j_q\rangle ,$$

que l'on peut réécrire en utilisant l'équation (5) :

$$\frac{1}{\sqrt{4}}\Big(|0\rangle + \omega_n^{j}|1\rangle\Big)\Big(|0\rangle + \omega_{2^{q-1}}^{j}|1\rangle\Big)|j_3\rangle \dots |j_q\rangle .$$

On continue de cette manière pour chacun des qubits, ce qui nous donne l'état :

$$\frac{1}{\sqrt{n}}\Big(|0\rangle + \omega_{2^q}^{j}|1\rangle\Big)\Big(|0\rangle + \omega_{2^{q-1}}^{j}|1\rangle\Big) \dots \Big(|0\rangle + \omega_2^{j}|1\rangle\Big).$$

On voit que l'on a obtenu la formule finale de l'équation (3), mais avec l'ordre des qubits inversé. C'est pourquoi nous avons les portes swap à la fin du circuit sur la figure 2.

Comptons maintenant le nombre de portes que nous avons utilisées. Le circuit est composé de q blocs, formés par les portes ciblant le même qubit. Le premier bloc, comportant toutes les portes ciblant le premier qubit donc, contient une porte Hadamard, ainsi que de $q-1$ portes de phase controlées, ce qui fait q portes. Le deuxième bloc contient une porte controlée de moins, donc $q-1$ portes, pour un total de $q + (q-1)$ portes. Continuant ainsi pour chaque qubit, on obtient

$$q + (q-1) + \dots + 1 = q(q+1)/2 \sim q^2$$

portes, plus les $q/2$ portes swap à la fin. Ce circuit décrit un algorithme quantique de complexité $\Theta(q^2)$ réalisant la transformée de Fourier discrète.

Du côté classique, la transformée de Fourier rapide, qui calcule également la transformée de Fourier discrète de n éléments, a une complexité $\Theta(n \log n) = \Theta(q2^q)$, ce qui est exponentiellement plus coûteux que la complexité de la transformée de Fourier quantique.

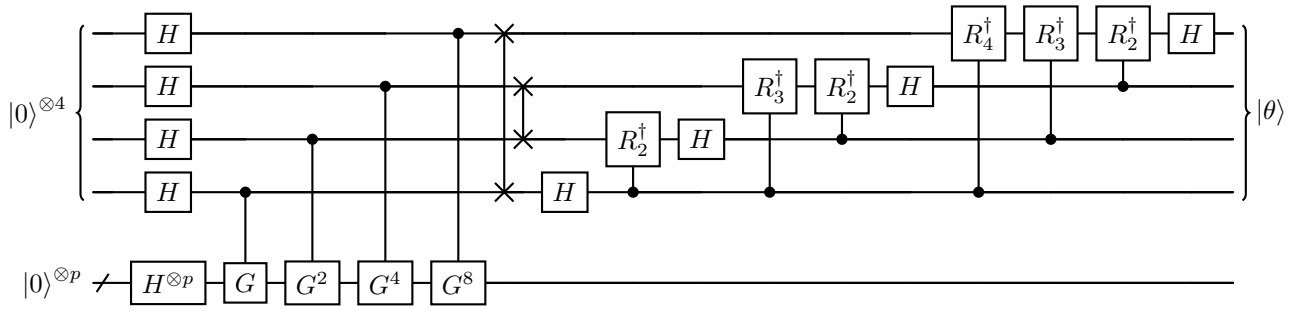

Figure 4: L'algorithme quantique de comptage appliqué à la recherche de Grover.

6 Application

L'algorithme d'estimation de phase peut être utilisée pour résoudre plusieurs problèmes intéressants, en particulier, il nous permet de connaitre le nombre optimal d'itérations pour l'algorithme de recherche de Grover, et est un ingrédient principal des algorithmes de recherche de période et de factorisation.

6.1 Algorithme de recherche de Grover

L'algorithme d'estimation de phase joue un rôle clé dans l'algorithme de recherche de Grover. En effet, pour appliquer l'algorithme de Grover efficacement, il est nécessaire de connaître le nombre de solutions à rechercher. Pour cela, on utilise l'algorithme de comptage, qui est une application directe de l'algorithme d'estimation de phase.

Considérons le problème suivant. On a un ensemble fini $\{0,1\}^q$ de taille $n = 2^q$, et on a un ensemble $B \subseteq \{0,1\}^q$ de solutions. Définissons la fonction f de la manière suivante :

$$f\colon \{0,1\}^q \longrightarrow \{0,1\}$$
$$x \longmapsto f(x) = \begin{cases} 1, & \text{si } x \in B, \\ 0, & \text{sinon.} \end{cases}$$

Le problème est le suivant : étant donnée la fonction f, déterminer la taille de $B = f^{-1}(1)$.

Pour le résoudre, on utilise l'algorithme d'estimation de phase, avec U l'opérateur de Grover. Il ne nous reste plus qu'à déterminer l'état initial du second registre. Définissons deux nouveaux états $|\alpha\rangle$ et $|\beta\rangle$ tels que

$$|\alpha\rangle = \frac{1}{\sqrt{n-|B|}} \sum_{x \notin B} |x\rangle\,, \quad \text{et} \quad |\beta\rangle = \frac{1}{\sqrt{|B|}} \sum_{x \in B} |x\rangle\,.$$

L'opérateur de Grover peut être interprété de la manière suivante : dans le plan généré par les deux états $|\alpha\rangle$ et $|\beta\rangle$, l'opérateur de Grover agit comme la rotation

$$G = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

et agit comme l'identité partout ailleurs. L'opérateur a donc deux valeurs propres différentes de 1, qui sont $e^{i\theta}$ et $e^{-i\theta}$. Les vecteurs propres associés à ces deux valeurs propres sont

$$|\varphi_+\rangle \coloneqq \frac{1}{\sqrt{2}}\Big(|\alpha\rangle + i\,|\beta\rangle\Big) \quad \text{et} \quad |\varphi_+\rangle \coloneqq \frac{1}{\sqrt{2}}\Big(|\alpha\rangle - i\,|\beta\rangle\Big)$$

On a en particulier que

$$\frac{1}{\sqrt{n}} \sum_{k=0}^{2^q-1} |k\rangle = \frac{1}{\sqrt{2}}\Big(\gamma\,|\varphi_+\rangle + \overline{\gamma}\,|\varphi_-\rangle\Big)$$

avec γ un scalaire de module 1. Cela nous permet de simplement initialiser le second registre dans l'état correspondant à la superposition uniforme de tous les états de la base computationelle, en appliquant une couche de porte Hadamard.

Comme $|\gamma| = 1$, à l'issue de l'algorithme d'estimation de phase, on mesure $\theta/2\pi$ ou $-\theta/2\pi$ avec égale probabilité, d'où on peut déduire θ ou $-\theta$. Comme sin est une fonction impaire, on a $\sin(\pm\theta) = \pm\sin\theta$. Par la suite, nous aurons seulement besoin de la valeur absolue de ce $\sin\theta$, donc n'importe quelle mesure à l'issue de l'algorithme d'estimation de phase nous convient.

Si l'on suppose que la taille de l'ensemble des solutions est moins grand que la moitié de la taille des solutions faisables, c'est-à-dire, si $2|B| \leq n$, alors l'analyse de l'algorithme de Grover nous dit que

$$\sin\frac{\theta}{2} = \sqrt{\frac{|B|}{n}}.$$

Ainsi, connaitre une des valeurs propres de l'opérateur de Grover nous donne le nombre de solutions à trouver, en supposant que l'entier n est connu. De plus, l'analyse de l'algorithme de Grover nous apprend également que le nombre d'itérations optimal est

$$\frac{\pi}{4}\sqrt{\frac{n}{|B|}}.$$

Donc, pour appliquer de façon optimale l'algorithme de recherche de Grover, il est utile d'utiliser l'algorithme de comptage en amont, afin de connaître $|B|$.

6.2 Recherche de période

Dans la suite, on va se focaliser sur les routines quantiques des algorithmes résolvant les problèmes de recherche de période et de factorisation. Ces algorithmes quantiques rapides pour résoudre ces problèmes sont intéressants pour au moins deux raisons. La première est que ces algorithmes nous donnent un exemple de problèmes qui peuvent être résolus de façon nettement plus efficace par un ordinateur quantique que par un ordinateur classique. La seconde raison est que ces algorithmes peuvent être utilisés pour casser le système de cryptographie à clé publique RSA.

Considérons deux entiers positifs x et n premiers relatifs tels que $0 < x < n$.

Définition 1. L'ordre de x modulo n est défini comme le plus petit entier strictement positif r tel que

$$x^r \equiv 1 \mod n,$$

c'est-à-dire $x^r - kn = 1$ pour un entier k.

Le problème de recherche de période consiste à déterminer l'ordre d'un élément x modulo n pour une paire donnée (x, n). Dans ce contexte, l'ordre de x modulo n est aussi appelée sa période. Pour résoudre ce problème, on va appliquer l'algorithme d'estimation de phase avec un opérateur unitaire U dont l'une des valeurs propres encode la période recherchée.

Exemple 2. Prenons $x = 5$ et $n = 21$. On a alors dans un premier temps que

$$x \equiv 5 \mod 21.$$

Considérons maintenant le carré de x, on a alors $x^2 = 25$. Cet entier est plus grand que 21, on peut alors soustraire 21 à x^2 pour obtenir un entier plus petit, mais équivalent modulo 21. On a alors

$$x^2 \equiv 4 \mod 21.$$

Multiplions une nouvelle fois par 5 pour obtenir

$$x^3 \equiv 20 \mod 21.$$

On répète ce processus jusqu'à obtenir 1, on a

$$x^4 \equiv 100 \equiv 16 \mod 21,$$

puis

$$x^5 \equiv 80 \equiv 17 \mod 21,$$

et enfin,

$$x^6 \equiv 85 \equiv 1 \mod 21.$$

L'ordre de $x = 5$ modulo $n = 1$ est $r = 6$. ◊

Sans entrer dans les détails, l'algorithme qui résout la recherche de période utilise un algorithme d'estimation de phase avec l'opérateur unitaire U défini par

$$U|y\rangle \coloneqq \begin{cases} |xy \bmod n\rangle, & \text{si } 0 \leq y \leq n-1, \\ |y\rangle, & \text{sinon.} \end{cases}$$

Proposition 6.1. *Les vecteurs propres*

$$\{|u_j\rangle \mid 0 \leq j \leq r-1\}$$

de l'opérateur U sont définis par :

$$|u_j\rangle \coloneqq \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \omega_r^{-jk} \left|x^k \bmod n\right\rangle.$$

Proof. On procède par calcul direct. On a

$$\begin{aligned} U|u_j\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \omega_r^{-jk} \left|x^{k+1} \bmod n\right\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{k=1}^{r} \omega_r^{-j(k-1)} \left|x^k \bmod n\right\rangle \\ &= \frac{\omega_r^j}{\sqrt{r}} \sum_{k=1}^{r} \omega_r^{-jk} \left|x^k \bmod n\right\rangle, \end{aligned}$$

et, parce que r est l'ordre de x modulo n, et par définition de ω_r, on a que

$$x^r \equiv 1 \equiv x^0 \mod n, \qquad \text{et} \qquad \omega_r^{-jr} = 1 = \omega_r^{-j0},$$

donc on peut remplacer le cas $k = r$ par $k = 0$ dans la somme ci-dessus, ce qui donne finalement

$$U|u_j\rangle = \frac{\omega_r^j}{\sqrt{r}} \sum_{k=0}^{r-1} \omega_r^{-jk} \left|x^k \bmod n\right\rangle = \omega_r^j |u_j\rangle.$$

□

Utiliser l'algorithme d'estimation de phase permet alors de calculer, avec une grande précision, la valeur propre ω_r^j correspondante, ce qui nous donne suffisamment d'information pour calculer la période r par la suite.

Supposons que l'on puisse appliquer efficacement les différentes puissances de l'opérateur U comme requis pour l'algorithme d'estimation de phase. La seule chose qui reste à construire pour pouvoir faire tourner l'algorithme de recherche de période est l'état initial du second registre, celui qui est la cible des portes controlées. Pour cela, on met le registre dans la superposition uniforme de tous les vecteurs propres de U.

Proposition 6.2. *On a l'égalité suivante :*

$$\frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} |u_j\rangle = |\underline{1}\rangle.$$

Proof. On procède encore une fois par calcul direct :

$$\begin{aligned} \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} |u_j\rangle &= \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} \left(\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \omega_r^{-jk} \left|x^k \bmod n\right\rangle \right) \\ &= \frac{1}{r} \sum_{k=0}^{r-1} \left(\sum_{j=0}^{r-1} \omega_r^{-jk} \right) \left|x^k \bmod n\right\rangle. \end{aligned}$$

Focalisons-nous sur le terme entre parenthèses. Utilisant de nouveau la formule des séries géométriques tronquées normalisée par r^{-1}, on a

$$\frac{1}{r}\sum_{j=0}^{r-1}\omega_r^{-jk} = \begin{cases} 0, & \text{si } \omega_r^{-k} \neq 1, \\ 1, & \text{sinon.} \end{cases}$$

Or, $\omega_r^{-k} = 1$ seulement quand $k = 0$. Branchons cela dans la formule précédente pour obtenir

$$\frac{1}{\sqrt{r}}\sum_{j=0}^{r-1}|u_j\rangle = \sum_{j=0}^{r-1}\omega_r^{-j0}\left|x^0 \bmod n\right\rangle = \left|x^0 \bmod n\right\rangle,$$

qui est effectivement égal à l'état $|\underline{1}\rangle$. □

Quelles sont les resources nécessaires pour pouvoir appliquer cet algorithme ? L'utilisation de l'algorithme d'estimation de phase sur q qubits, ce qui requiert $O(q^2)$ portes pour la transformée de Fourier inverse. Nous ne l'avons pas précisé plus tôt, mais il existe un algorithme pour appliquer successivement les différentes puissances controlées de l'opérateur d'exponentiation U, qui requiert $O(q^3)$ portes. Enfin, il existe également un algorithme de fractions continues pour déduire la valeur de r des différents samples obtenus, qui requiert également $O(q^3)$ portes. La complexité globale de l'algorithme de recherche de période est donc de l'ordre de

$$O\big(q^3\big) = O\Big(\big(\log n\big)^3\Big) \text{ portes.}$$

Pour conclure cette partie sur l'algorithme de Shor, on va montrer rapidement comment le problème de factorisation peut être résolu grâce à l'algorithme de recherche de période. Supposons que nous ayons un nombre n qui n'est pas premier. Le but est de connaitre l'un de ses diviseurs. On peut répéter cette procédure plusieurs fois jusqu'à obtenir tous les diviseurs.

Prenons un entier $1 \leq x \leq n-1$ au hasard, uniformément. Si le plus grand diviseur commun est plus grand que 1, on renvoie directement cette valeur et on recommence. Si x et n sont premiers relatifs, alors on calcule la période r de x modulo n, grâce à l'algorithme quantique de recherche de période. Si r est impair, ou si $x^{\frac{r}{2}} \equiv -1 \mod n$, on abandonne et on choisit un nouveau x aléatoirement. Si r est pair et $x^{\frac{r}{2}} \not\equiv -1 \mod n$, alors on calcule le plus grand diviseur commun de la paire $(x^{\frac{r}{2}} - 1, n)$ ainsi que de la paire $(x^{\frac{r}{2}} + 1, n)$. Il se trouve qu'au moins l'une de ses deux valeurs est nécessairement un diviseur de n, que l'on renvoie. L'efficacité de cet algorithme dépend du résultat suivant, que nous n'allons pas démontrer ici.

Théorème 6.1. *Supposons que N est le produit de m nombres premiers distincts, et que x est tiré aléatoirement uniformément dans $\{1, \ldots, n-1\}$ et que x et n sont premiers relatifs. Notons r l'ordre de x modulo n. Alors la probabilité que r soit pair et que $x^{\frac{r}{2}} \not\equiv -1 \mod n$ est au moins $1 - 2^{-m}$.*