INF581: Information Quantique **Hiver 2010**

Cours 9 — 20 avril

Enseignant : Philippe Grangier – Frédéric Magniez *Rédacteur : Issam El Alaoui*

9.1 Algorithme de Kitaev et marches quantiques

9.2 Estimation de phase

9.2.1 Description

Nous avons vu dans le cours précédent comment il était possible grâce à la transformée de fourrier discrète quantique sur $\mathbf{Z}^{2n}$ d'obtenir la somme $\frac{1}{\sqrt{N}} \sum_y w^{xy} |y\rangle$ en utilisant un circuit quantique de taille $(\log N)^2$. Considérons à présent le problème inverse

Problème

En partant cette fois-ci d'un état du type $\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{i\alpha y} |y\rangle$ avec un angle $0 \leq \alpha \leq 2\pi$

Pouvons-nous obtenir en sortie les n premiers bits de $\frac{\alpha}{2\pi}$?

Solution

Le cas simple Lorsque $\alpha = 2\pi \frac{x}{2^n}$, il suffit d'utiliser la transformée de Fourier inverse pour avoir x.

Le cas général La transformée de Fourier inverse donne x tel que les $m = n+1+\log(1+\frac{1}{\epsilon})$ premiers bits de x vérifient

$$\Pr\left(\frac{x'}{2^m} - \alpha \leq \frac{1}{2^m+1}\right) \geq 1 - \epsilon$$

Application

L'application naturelle de la résolution de se problème se fait sur l'estimation de valeurs propres.

En effet, considérons en entrée des boites noires réalisant $c - U^t$ pour U transformation unitaire, et $t = 1, 2...2^n$
Considérons également ϕ un vecteur propre de U pour la valeur propre $e^{i\alpha}$.

On obtient alors en sortie $\frac{\alpha}{2\pi}$ à n bits de précision près.

9.2.2 Circuit

La circuit résolvant le problème est constitué des portes U^{2^i} qui déphasent l'état voulu et le contrôlent par rapport à un autre état de référence qui est inchangé.

Portes

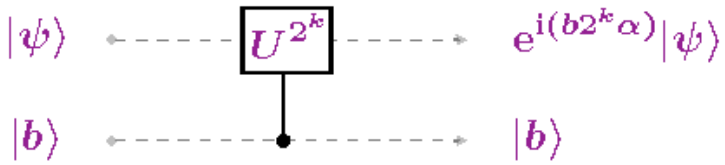

Figure 9.1. Portes utilisées

Circuit complet

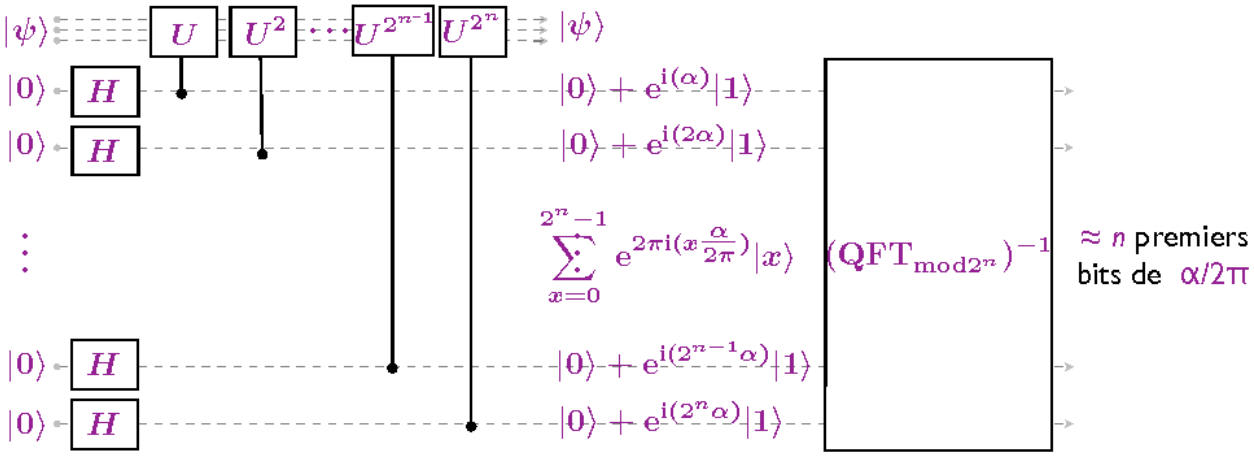

Figure 9.2. Circuit complet

9.3 Outils théoriques

9.3.1 Marches aléatoires

Définition

Définition 9.1. *Considérons un graphe non orienté $G = (V, E)$ sur les sommets V et d'arêtes E.*
La marche aléatoire est un déplacement aléatoire sur les sommets V de G en suivant les arêtes E de G.
La probabilité d'aller d'un sommet u à un sommet v est égale à l'inverse du degré du sommet de départ, le degré d'un sommet étant le nombre d'arêtes allant de ce sommet.
C'est-à-dire

$$\Pr(u \to v) = \frac{1}{deg(u)}$$

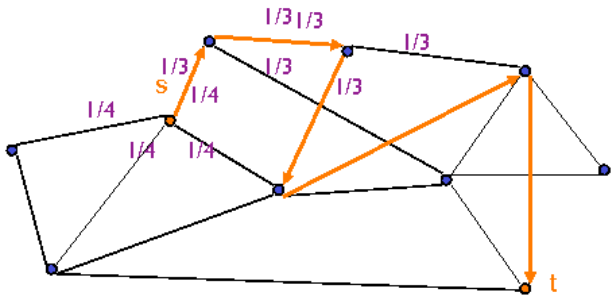

Figure 9.3. Marche aléatoire

9.3.2 Applications

Les marches aléatoires servent concrêtement à :
- Trouver les chemins aller d'un sommet s à un sommet t.
- Mélanger un jeu de cartes de manière efficace, et trouver le temps pour effectuer le mélange.
- Faire des estimations de surface, de volume, des comptages (fameux Page rank de Google)
- Parcourir des graphes le plus rapidement possible
- Prédire le comportement d'une particule aléatoire
- Modéliser et analyser des comportements (réseaux, jeux, finance).
- Chercher une information dans des bases de données immenses.

9.3.3 Chaîne de Markov

Définition 9.2. *Processus stochastique*
Un processus stochastique (discret à espace fini) $\boldsymbol{X}$ est une suite de variables aléatoires X_t

indexée par un paramètre temps (discret, $t = 0, 1, 2, 3...$)
Espace fini désigne un espace dans lequel les X_t prend un nombre fini de valeurs.

Définition 9.3. *Chaîne de markov*
Une chaine de markov est un processus stochastique qui vérifie :

$$\begin{aligned}\Pr[X_t = a_t | X_{t-1} = a_{t-1}, X_{t-2} = a_{t-2}, ..., X_0 = a_0] \\ = \Pr[X_t = a_t | X_{t-1} = a_{t-1}] \\ = P(a_t, a_{t-1})\end{aligned}$$

Cela signifie qu'il n'y a pas d'effet mémoire, puisqu'il y a indépendance entre la valeur prise en t et les valeurs précédents $t-1$, mais également que le processus est indépendant du temps. En résumé, l'évolution ne dépend que de la valeur courante.
Les matrices de transition décrivent bien les chaînes de markov, puisqu'on a d'après les deux propriétés énoncées précédemment :

$$P(i, j) = \Pr[X_t = i]$$

La probabilité d'être dans l'état i en X_t est $P_t(i) = \Pr[X_t = i]$ P_t étant $P_0 P^t$

Exemples

Si P est une marche aléatoire (donnée par sa matrice de transitions), une marche aléatoire convexe est par exemple :

$$Q = \frac{1}{2}P + \frac{1}{2}I_d$$

Une somme convexe de marches aléatoires est par exemple :

$$R = \frac{3}{4}P + \frac{1}{4}Q$$

Propriétés

Définition 9.4. *Une distribution est dite **stationnaire** si c'est une distribution de probabilité π telle que $\pi = \pi * P$*

Définition 9.5. *Une marche aléatoire est dite **ergodique** si elle est définie sur un graphe*
- *connexe, c'est-à-dire que tous les sommets sont connectés au graphe*
- *non biparti, ce qui implique que les arêtes ne sont pas uniquement entre deux sous-parties disjointes*

Remarque
L'ergodicité se généralise aux chaines de Markov

Théorème 9.6. *Toute chaine de Markov ergodique possède une unique distribution stationnaire π*

Temps de mélange

Définition 9.7. *On définit le temps de mélange de la manière suivante*

$$Max_x\{min_T|P^T(x,i) \approx \pi_i\}$$

On appelle **Spectral gap** $\delta = 1 - 2$*plus grande valeur propre*
Le temps de mélange d'une marche aléatoire est en $O(\frac{1}{\delta}\log(\frac{1}{min_x \pi_x}))$. Cette propriété se généralise aux chaînes de Markov réversibles.

9.3.4 Marches quantiques

Les marches aléatoires peuvent se généraliser au cas quantique. On parle alors de marches sur les bords.
Dans le cas où nous disposons d'un graphe G avec n sommets connectés, nous pouvons définir la transition $x \to y$ grâce à la probabilité $P_{x,y}$ de transition d'un sommet à l'autre.
On définit alors P l'application de transition de x vers l'état de destination y qui met x le résultat dans un état de superposition de x et de y.

$$P = (P_{x,y}, ..., P_{x,y})$$

Il faut ensuite échanger le rôle de l'origine et de la destination via un opérateur *Swap*.
L'opérateur global obtenu est alors :

$$Swap \cdot \left(\sum_x |x\rangle \langle x| \otimes F^x \right)$$

9.4 Analyse d'un algorithme - SAT Problem

9.4.1 Définition

SAT

Le problème SAT (Satisfiability) consiste à essayer de résoudre une équation logique, c'est-à-dire vérifier la satisfiabilité d'une telle équation. Etant donné en entrée un ensemble de clauses sur n variables, on cherche une assignation sur ces variables qui vérifie toutes ces clauses.
Exemple

$$x_1 \vee x_3 \vee \overline{x1} \wedge x_2 \vee \overline{x_4} \wedge \overline{x_2}$$

Le problem k-SAT est le problème SAT pour lequel on se restreind à seulement k variables distinctes. On montre par le *théorème de Cook* que n-SAT peut être ramené à 3-SAT.

Théorème 9.8. *3-SAT est un problème NP-Complet (résoluble par une machine de Turing non détérministe en temps polynomial)*
2-SAT est résoluble en temps et espace linéaire [http ://dx.doi.org/10.1016/0020-0190(79)90002-4]
2-SAT est résoluble en temps quadratique et espace logarithmique

9.4.2 2-SAT

Nous présentons un algorithme pour 2-SAT en espace logarithmique

Algorithme

Le procédé est le suivant :

1. Choisir une assignation a quelconque dans $\{0,1\}^n$
2. Répéter $2mn^2$ fois, en s'arrêtant si on tombe sur la satisfiabilité
3. – Choisir une clause C arbitrairement qui soit non satisfaite
   – Choisir aléatoirement une variable x_i de C
   – Changer la valeur du bit i de a correspondant à cette variable
4. Si a satisfait toute les clauses, renvoyer la satisfiabilité, sinon renvoyer la non satisfiabilité.

Analyse de l'algorithme

Considérons une éventuelle assignation solution s. Soit X_t le nombre de bits différents entre a et s à la t-ème itération.
Alors, nécessairement :

$$\Pr[X_{t+1} = n-1 | X_t = n] = 1$$
$$\Pr[X_{t+1} = 0 | X_t = 0] = 1$$
$$\Pr[X_{t+1} = j-1 | X_t = j] \geq \frac{1}{2}$$

Version pessimiste
Nous partons de l'état le plus défavorable

$$Y_0 = n$$

$$\Pr[Y_{t+1} = n-1 | Y_t = n] = 1$$
$$\Pr[Y_{t+1} = 0 | Y_t = 0] = 1$$
$$\Pr[Y_{t+1} = j-1 | Y_t = j] = \frac{1}{2}$$

Introduisons à présent h_j, le temps moyen pour atteindre l'état 0 en allant de l'état j, alors :

$$h_n = 1 + h_{n-1}$$

$$h_0 = 0$$
$$h_j = \frac{h_{j-1} + h_{j+1}}{2} + 1$$

Cette suite à récurrence linéaire se résout pour $j \neq 0$ en :

$$h_j = h_{j-1} + 2(n-j) + 1$$

D'où :

$$h_n = \sum i = 0n(2i+1) = n^2$$

Enfin, en utilisant l'inégalité de Markov, on conclut que la probabilité de ne pas trouver de solutions apres $2n^2$ étapes est au plus un demi :

$$\Pr(T \geq 2n^2) \leq \frac{E(T)}{2n^2}$$

Soit, puisque $E(T) \leq h_n$

$$\Pr(T \geq 2n^2) \leq \frac{1}{2}$$

Après $2mn^2$ étapes, la probabilité de ne pas avoir de solution est au plus $\frac{1}{2^m}$

9.4.3 3-SAT

Algorithme

Le même algorithme est utilisé pour le cas 3-SAT que pour le cas 2-SAT. Malheureusement, celui-ci a tendance à s'éloigner de la solution, comme le montre le schéma suivant. La probabilité d'avancer dans la bonne direction est 1/3, il faut donc partir d'assignations

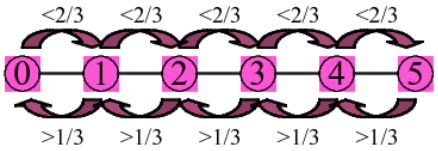

Figure 9.4. 3-SAT

aléatoires et itérer 3n fois, et recommencer.

Théorème 9.9. *L'algorithme ainsi modifié trouve une solution en un temps moyen de* $\left(\frac{4}{3}\right)^n$

Ce résultat est meilleur que toutes les approches détérministes dans le pire des cas. Toutes les marches aléatoires pour 3-SAT suivent la même structure.

9.5 Exercices

9.5.1 Opérateur Multiplication

Soit a un entier module N, premier avec N, d'ordre r.
Soit la transformation : $U_a : |x\rangle \rightarrow |ax\rangle$

Montrer que $(U_a)^t = U_{a^t}$

Solution
Soit x quelconque, considérons :

$$U_{a^t} |x\rangle = \left|a^t x\right\rangle = (U_a)^t |x\rangle$$
$$U_{a^t} = U_a^t$$

En déduire que les valeurs propres λ de U_a satisfont $\lambda^r = 1$
Solution
Etant donné que a est d'ordre r modulo N

$$U_{a^r} = I_d$$

Donc en écrivant l'égalité pour tout vecteur $|x\rangle$, on obtient $\lambda^r = 1$

Montrer que chaque vecteur $|\psi_k\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} w_r^{jk} |a^j >$ est vecteur propre de U_a pour une valeur propre à calculer.
Solution
Appliquons U_a à chaque vecteur $|\psi_k\rangle$

$$U_a |\psi_k\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} w_r^{jk} |a^{j+1} >$$
$$= \frac{w^{-k}}{\sqrt{r}} \sum_{j=1}^{r} w_r^{jk} |a^j >$$
$$= \frac{w^{-k}}{\sqrt{r}} \sum_{j=0}^{r-1} w_r^{jk} |a^j >$$

Etant donné que w_r est racine r-ème de l'unité, et que a est d'ordre r, on déduit que l'élément de la somme d'indice r est égal à celui d'indice 0, d'où la simplification.
$|\psi_k\rangle$ est bien vecteur propre de U_a de valeur propre w^{-k}.

9.5.2 Factoriser

Montrer qu'en partant d'un superposition uniforme de vecteurs propres $|\psi_k >$ de U_a, alors l'estimation de valeur propre avec m bits de précision fourni y tel quel $\frac{y}{2^m} \approx \frac{k}{r}$
Solution
Puisque $|\psi_k\rangle$ est vecteur propre de U_a de valeur propre w_r^{-k}, l'estimation de valeur propre fournit, en utilisant des boites $c - U_a^t$, la valeur $-2\pi\frac{k}{r}$.

Quelle est la superposition uniforme des vecteurs propres $|\psi_k >$ de U_a ?.
Solution
Les $|\psi_k >$ se rapporte en fait aux racines de l'unité et, comme dans le cas complexe, leur superposition est égale à $|1 >$.

Conclure avec un autre algorithme pour la factorisation. Donner sa complexité.
Solution On en déduit donc que l'état $|1 >$ est une superposition d'états solutions et que donc, une simple mesure sur cet état donnera une solution au problème de la factorisation.