École Centrale de Lille - Université de Lille

Equipe Projet INRIA PARADYSE Lille - Lab. P.Painleve CNRS

Vers l'avantage quantique ?

Auteur :
Victor Niaussat
Encadrant :
Stephan De Bièvre

28 février 2023

Table des matières

1 Introduction

Les technologies quantiques espèrent mettre à profit les différences entre la mécanique classique et quantique afin d'exécuter de façon plus rapide, plus efficace ou plus sûre un certain nombre de tâches. C'est ce qu'on appelle l'avantage quantique. Le débat reste ouvert de savoir dans quelle mesure un tel avantage peut être atteint, dans quelles circonstances et pour quel type de tâche. Le but de ce mémoire est de se familiariser avec cette problématique dans le contexte du quantum random sampling et/ou du machine learning quantique.

Après s'être familiarisée avec la problématique, l'étudiant adressera quelques-unes des questions mathématiques qui se présentent naturellement dans ce contexte. Le stage vient en complément naturel au cours sur l'information quantique du S4.

Le sujet du stage est typiquement un sujet de physique mathématique, et se situe à l'intersection de l'analyse fonctionnelle, des probabilités et de la modélisation. Comme plusieurs questions restent ouvertes, des simulations numériques sont intéressantes et peuvent faire partie du stage, selon les goûts de l'étudiant.

2 Boson Random Sampling

Le Boson random samplingest une variante de l'algorithme de quantum computing qui utilise des photons (particules de lumière) pour effectuer des calculs. Cet algorithme a été introduit pour la première fois en 2011 par Scott Aaronson et Alex Arkhipov [**aaronson_computational_2013**], qui ont montré qu'il était possible d'utiliser un réseau de photons pour effectuer des calculs qui sont difficiles à effectuer de manière classique, même avec des ordinateurs quantiques.

Le Boson random samplingest basé sur le principe de superposition quantique, qui permet aux photons de se trouver à plusieurs endroits à la fois. Cela signifie qu'un photon peut suivre plusieurs chemins différents en même temps, ce qui peut être utilisé pour effectuer des calculs de manière plus efficace que de manière classique. De plus, de nouveaux algorithmes ont été mis en oeuvre à partir du Boson random samplingpour réaliser des taches irréalisable avec un ordinateur classique (des générateurs sans biais de nombre aléatoires , un algorithme de calculs de permanent de matrices définie semi positive [**chakhmakhchyan_quantum-inspired_2017**] ...).

Il a suscité un grand intérêt dans le domaine de la computation quantique, car il offre un moyen de démontrer la supériorité de la computation quantique sur la computation classique pour certains types de calculs. Cependant, l'algorithme est également très difficile à mettre en œuvre de manière pratique, car il nécessite une grande quantité de photons et un grand nombre de fonctions d'onde pour être exécuté correctement.

En résumé, le Boson sampling est un algorithme de quantum computing qui utilise des photons pour effectuer des calculs de manière plus efficace que de manière classique, mais qui est également très difficile à mettre en œuvre de manière pratique.

2.1 Espace de Fock , opérateur de création et d'anhilation

En mécanique quantique, un état de Fock ou état numérique est un état quantique qui est un élément d'un espace de Fock avec un nombre bien défini de particules (ou quanta).

On spécifie un état multiparticulaire de N particules identiques non interactives en écrivant l'état comme une somme de produits tensoriels de N états à une particule. De plus, selon l'intégralité du spin des particules, les produits tensoriels doivent être des produits alternatifs (antisymétriques) ou symétriques de l'espace de Hilbert sous-jacent à une particule. Plus précisément :

- Les fermions, possédant un spin demi-entier et obéissant au principe d'exclusion de Pauli, correspondent à des produits tensoriels antisymétriques.
- Les bosons, possédant un spin entier (et non régis par le principe d'exclusion) correspondent à des produits tensoriels symétriques.

Dans ce mémoire, nous nous concentrerons sur les bosons et plus particulièrement sur les photons. Ce sont les particules qui composent la lumière et avec lesquelles on peut travailler. C'est le champs de l'optique quantique.

2.1.1 Etat de Fock

Un état de Fock s'écrit alors comme ceci : donnons $(s_i)_{i\in I}$ une base orthonormal d'état dans un espace de Hilbert sous-jacent à 1 particule.

Cela induit une base correspondante de l'espace de Fock appelée "base du nombre d'occupation". Un état quantique dans l'espace de Fock est appelé état de Fock s'il est un élément de la base du nombre d'occupation.

Un état de Fock satisfait un critère important : pour chaque i, l'état est un état propre de l'opérateur nombre de particules N_{s_i} correspondant au ième élément de l'état de Fock représentant l'état s_i.

Un état quantique dans l'espace de Fock peut s'écrire comme ceci : les $m_{s_i} \in \mathbb{N}$ représente le nombre de particule dans l'état s_i , et on écrit cette état : $|m_{s_1} \dots m_{s_n}\rangle$.

Ainsi, l'opérateur N_{s_i} agit comme ceci sur l'état de Fock :

$$N_{s_i}|m_{s_1} \dots m_{s_n}\rangle = m_{s_i}|m_{s_1} \dots m_{s_n}\rangle$$

2.1.2 Opérateur créateur et annhilateur pour les bosons

Les opérateurs de création et d'annihilation sont des opérateurs mathématiques qui ont de nombreuses applications en mécanique quantique, notamment dans l'étude des oscillateurs harmoniques quantiques et des systèmes à plusieurs particules.

Un opérateur d'annihilation (généralement noté a) diminue d'une unité le nombre de particules dans un état donné. Un opérateur de création (généralement noté $a^\dagger$) augmente de un le nombre de particules dans un état donné, et il est l'adjoint de l'opérateur d'annihilation.

Ainsi , $\forall|m_{s_1} \dots m_{s_n}\rangle$, état de l'espace de Fock,

$$a_{s_i}|m_{s_1} \dots m_{s_i} \dots m_{s_n}\rangle = \sqrt{m_{s_i}}|m_{s_1} \dots m_{s_i} - 1 \dots m_{s_n}\rangle$$
$$a_{s_i}{}^\dagger|m_{s_1} \dots m_{s_i} \dots m_{s_n}\rangle = \sqrt{m_{s_i}+1}|m_{s_1} \dots m_{s_i} + 1 \dots m_{s_n}\rangle$$

Ce ne sont pas des observables et on peut définir l'opérateur nombre de particule :

$$N_{s_i} = a^\dagger_{s_i} a_{s_i}$$

Les relations de commutation des opérateurs de création et d'annihilation dans un système à bosons multiples sont :

$$[a_i, a_j^\dagger] \equiv a_i a_j^\dagger - a_j^\dagger a_i = \delta_{ij}$$

$$[a_i^\dagger, a_j^\dagger] = [a_i, a_j] = 0$$

2.2 Element d'optique quantique

En général, pour réaliser unc ordinateur quantique, nous avons besoin d'un moyen de préparer des états quantiques, d'effectuer un ensemble de transformation sur les qubits grâce à des portes quantiques universelles et de mesurer l'état de sortie. Pour générer un état quantique, nous utilisons une source de photons unique qui ajoute un photon à l'état de vide $|0\rangle$ et fait ainsi passer tout mode de vide à l'état $|1\rangle$. Ce processus est non déterministe mais il est suffisant pour le calcul quantique. Les éléments optiques les plus simples sont les déphaseurs (*phase-shifters*) et les séparateurs de faisceaux (*beamsplitters*). Ces éléments sont utilisés pour agir comme des opérations de porte sur nos états préparés. Comme ces deux transformations sont unitaires, nous pouvons écrire chacun de ces éléments en termes de matrice unitaire. Une matrice unitaire de déphaseur, agissant sur un seul mode avec N est l'opérateur nombre de particule, est simplement :

$$P_\phi = e^{iN\phi}$$

La matrice unitaire d'un diviseur de faisceau est donnée par

$$B_{\theta,\phi} = \begin{pmatrix} cos\theta & -e^{i\phi}sin\theta \\ e^{i\phi}sin\theta & cos\theta \end{pmatrix}$$

dans la base des modes optiques. θ représente le biais du séparateur et ϕ donne la relation de phase

2.3 Boson random sampling

Considérons un circuit optique linéaire multimode de m modes qui est injecté avec n photons uniques indiscernables ($m > n$). La mise en œuvre photonique de l'échantillonnage de bosons consiste à générer un échantillon à partir de la distribution de probabilité des mesures de photon unique en sortie du circuit. Plus précisément, cela nécessite des sources fiables de photons uniques , ainsi qu'un interféromètre linéaire. Ce dernier peut être fabriqué, par exemple, avec des séparateurs de faisceau à fibres fusionnées, par silice sur silicium ou écrit au laser des interféromètres intégrés, ou des puces optiques à interface électrique et optique.

Enfin, le schéma nécessite également des détecteurs de comptage de photons uniques à haut rendement, tels que ceux basés sur des nanofils supraconducteurs polarisés en courant , qui effectuent les mesures à la sortie du circuit. Par conséquent, sur la base de ces trois ingrédients, la configuration d'échantillonnage de bosons ne nécessite aucun bit auxilliaire, mesure adaptative ou opération d'intrication. Cela en fait un modèle non universel de calcul quantique et réduit la quantité de ressources physiques nécessaires à sa réalisation pratique

En considérant a_j l'opérateur d'annihiliation (*resp* $a_j^\dagger$ l'opérateur de création), l'interféromètre réalise une transformation linéaire de l'opérateur de création :

$$b_j^\dagger := UajU^\dagger = \sum_{i=1}^{m} U_{ji} a_i^\dagger$$

Ainsi, on observe qu'à la sortie du circuit linéaire une densité de probabilité dépendant du circuit linéaire U. U est une matrice unitaire.

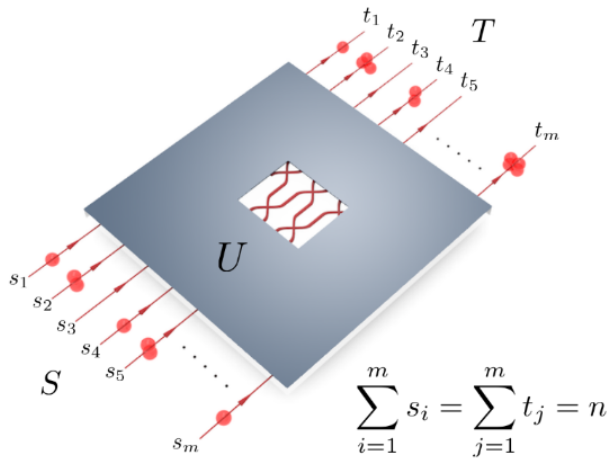

FIGURE 1 – Circuit du Boson random sampling

Il y a conservation du nombre de photons.

Ainsi, le circuit linéaire optique réalise une transformation linéaire de l'opérateur de création. Pour un état $|\psi_{in}\rangle$ d'entrée, on obtient un état de sortie $|\psi_{out}\rangle$ avec un opérateur unitaire $V := \phi(U)$ relié à U qui agit sur les états :

$$|\psi_{out}\rangle = V|\psi_{in}\rangle = \phi(U)|\psi_{in}\rangle$$

ψ possède 2 propriétés :

— Pour U unitaire, $\phi(U)$ est une transformation unitaire

— ϕ est un homomorphisme des matrices unitaires *i.e* $\forall U, V$ matrices unitaires, $\phi(UV) = \phi(U)\phi(V)$

2.4 Analogie avec la planche de Galton

Le Boson Random Sampling est similaire au *Galton Board*.

La table de Galton est un dispositif inventé par Sir Francis Galton qui illustre la convergence de la loi binomiale avec la loi normale.

Les clous sont enfoncés sur la planche de sorte qu'une boule lâchée sur la planche aille à droite ou à gauche pour chaque rangée de clous. En bas, les balles sont regroupées par le nombre de passes gauche et droite effectuées.

Ainsi, chaque case correspond à une issue possible de l'expérience binomiale (comme une expérience de Bernoulli répétée) et on voit que la répartition des boules dans les cases se rapproche d'une courbe gaussienne, d'autant plus que le nombre de lignes augmente ; autrement dit : la distribution binomiale converge vers la distribution normale. Ceci est donc une illustration du théorème de Moivre-Laplace.

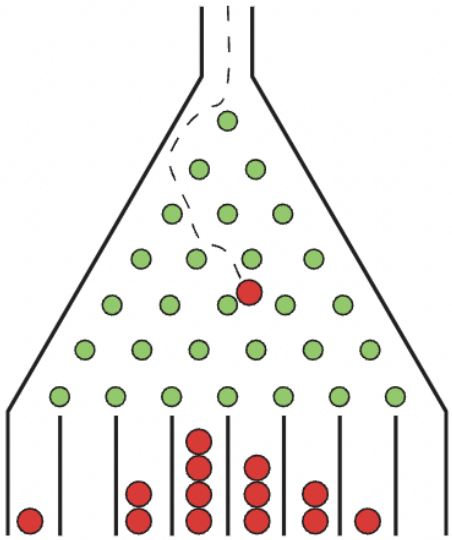

FIGURE 2 – Galton Board

2.5 Probabilité de sortie : apparition du permanent

La probabilité de mesurer $|t_1 t_2 \dots t_m\rangle$ avec une entrée $|s_1 s_2 \dots s_m\rangle$ dans la transformation unitaire U est :

$$P_U(S,T) = |\langle t_1 t_2 \dots t_m | \psi_{out} \rangle|^2 = \frac{|\mathrm{Perm}(U_{S,T})|^2}{\prod_{j=1}^{m}(s_j!)\prod_{i=1}^{m}(t_i!)}$$

avec

$$\mathrm{Perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma_i}.$$

La principale raison de l'intérêt croissant pour le modèle d'échantillonnage de bosons est que, malgré son caractère non universel, on pense fortement qu'il permet d'effectuer une tâche de calcul impossible à réaliser par un ordinateur classique mais peut être par un ordinateur quantique réel. L'une des principales raisons en est que la distribution de probabilité, que le dispositif d'échantillonnage de bosons doit échantillonner, est liée comme on peut le voir au permanent des matrices complexes.

Le calcul du permanent est, dans le cas général, une tâche extrêmement difficile : il tombe dans la classe de complexité #P-hard. De plus, son approximation à l'erreur multiplicative près est également un problème #P-hard.

Ainsi, en échantillonant plusieurs fois un circuit de Boson sampling, on peut s'attendre à approximer la densité de probabilité créée par l'opérateur U du circuit optique. Néanmoins, bien que la probabilité $P_U(S,T)$d'un résultat de mesure spécifique à la sortie de l'interféromètre est liée au permanent d'une sous matrice de U, une matrice unitaire, une machine d'échantillonnage de bosons ne permet pas son estimation. La raison principale est que la probabilité de détection correspondante est généralement exponentiellement faible. Ainsi, afin de collecter suffisamment de statistiques pour approximer sa valeur, il faut exécuter l'expérience quantique pendant une durée exponentiellement longue. Par conséquent, l'estimation obtenue à partir d'un échantillonneur de bosons n'est pas plus efficace que l'exécution de l'algorithme classique.

Ce qu'il faut bien comprendre, c'est que malgré le fait que le Boson random samplingn'est pas plus efficace pour calculer un permanent, il est cependant efficace pour générer une distribution irréalisable (sous réserve de $P \neq NP$) sur un ordinateur classique. Sans le détail complet, Aaronson et Arkhipov ont montré que si un algorithme classique peut échantilloner des états de Fock à partir de la même distribution de probabilité que peut "générer" le Boson sampling, $P = NP$.

2.5.1 Positionnement du problème

Pour réécrire un état du système avec s_i photons à l'entrée i, sachant qu'il y'a n photons et m modes, on décrit ce vecteur d'entrée $|\psi_{in}\rangle = |s_1 s_2 \dots s_m\rangle$.

Dans l'état de départ, il y a n photons. Ainsi, l'espace de départ est :

$$\Phi_{m,n} = \{(s_1, s_2, \dots, s_m) : \sum_{j=1}^{m} s_j = n\}$$

Cette espace est stable par une transformation unitaire : il y a conservation du nombre de photons par transformation unitaire. En effet, dans le modèle, on considère qu'il y a ni création ni destruction de photons après la passage de l'état dans le circuit optique.

On peut considérer le circuit linéaire optique comme un opérateur unitaire V qui s'applique à un état d'entrée :

$$|\psi_{out}\rangle = V|\psi_{in}\rangle$$

Ce qui nous intérésse est la probabilité de sortie

$$P_U(\text{in } = |\psi_{in}\rangle, \text{out } = \langle t_1 t_2 \dots t_m|)\rangle = |\langle t_1 t_2 \dots t_m|\psi_{out}\rangle|^2$$

Or

$$\langle t_1 t_2 \dots t_m|\psi_{out}\rangle = \langle t_1 t_2 \dots t_m|V|\psi_{in}\rangle = \langle 0| \prod_{i=1}^{m} \frac{a_i^{t_i}}{\sqrt{t_i!}} V|\psi_{in}\rangle$$

Si on a zéro photon à l'entrée du circuit, on a zéro photon à la sortie et donc $V|0\rangle = |0\rangle$ et $V^\dagger|0\rangle = |0\rangle$.

De plus, comme $V^\dagger V = I$, on peut rajouter l'opérateur V et $V^\dagger$ de part et d'autre des coefficients multiplicatifs. Ils se s'annule en opérateur identité dans les multiplications et on a :

$$V^\dagger \left(\prod_{i=1}^{m} \frac{a_i^{t_i}}{\sqrt{t_i!}} \right) V = \prod_{i=1}^{m} \frac{(V^\dagger a_i V)^{t_i}}{\sqrt{t_i!}}$$

Ainsi , en définissant $b_j := V^\dagger a_j V$ et $|\phi\rangle = \prod_{i=1}^{m} \frac{(b_i^\dagger)^{t_i}}{\sqrt{t_i!}}|0\rangle$:

$$= \langle 0|VV^\dagger \left(\prod_{i=1}^{m} \frac{a_i^{t_i}}{\sqrt{t_i!}} \right) V|\psi_{in}\rangle = \langle 0| \prod_{i=1}^{m} \frac{(V^\dagger a_i V)^{t_i}}{\sqrt{t_i!}} |\psi_{in}\rangle = \langle 0| \prod_{i=1}^{m} \frac{b_i^{t_i}}{\sqrt{t_i!}} |\psi_{in}\rangle = \langle \phi|\psi_{in}\rangle$$

Par conséquent, d'après la définition du modèle du Boson random sampling, cela nous revient à calculer :

$$|\phi\rangle = \prod_{j=1}^{m} \frac{1}{\sqrt{t_j!}} (\sum_{i=1}^{m} U_{ji} a_i^\dagger)^{t_j} |0\rangle$$

2.5.2 Pourquoi le permanent ? Exemple avec 2 photons

Soit $U = \begin{pmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{pmatrix}$ matrice représentant le circuit.
On suppose que U est unitaire.

$$b_j^\dagger = U_{j1}a_1^\dagger + U_{j2}a_2^\dagger$$

Ainsi :

$$|11\rangle = b_1^\dagger b_2^\dagger |00\rangle = (U_{11}a_1^\dagger + U_{12}a_2^\dagger)(U_{21}a_1^\dagger + U_{22}a_2^\dagger)|00\rangle$$
$$= (U_{11}U_{21}a_1^{\dagger 2} + U_{11}U_{22}a_1^\dagger a_2^\dagger + U_{12}U_{21}a_2^\dagger a_1^\dagger + U_{12}U_{22}a_2^{\dagger 2})|00\rangle$$

On sait que $[a_i^\dagger; a_j^\dagger] = 0$

$$= \sqrt{2}U_{11}U_{21}|20\rangle + (U_{11}U_{22} + U_{12}U_{21})|11\rangle + \sqrt{2}U_{12}U_{22}|02\rangle$$

D'après la règle de Born, on obtient les résultats de probabilités de sorties du circuit :

$$|\langle 02|11\rangle|^2 = 2\,|U_{11}|^2\,|U_{21}|^2 = \frac{1}{2}\left|\text{Perm}\begin{pmatrix} U_{11} & U_{11} \\ U_{21} & U_{21} \end{pmatrix}\right|^2$$

$$|\langle 20|11\rangle|^2 = 2\,|U_{12}|^2\,|U_{22}|^2 = \frac{1}{2}\left|\text{Perm}\begin{pmatrix} U_{21} & U_{22} \\ U_{21} & U_{22} \end{pmatrix}\right|^2$$

$$|\langle 11|11\rangle|^2 = \left|\text{Perm}\begin{pmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{pmatrix}\right|^2$$

2.5.3 Preuve du permanent

Partons désormais du fait que nous avons à calculer :

$$|\phi\rangle = \prod_{j=1}^{m} \frac{1}{\sqrt{t_j!}} (\sum_{i=1}^{m} U_{ji}a_i^\dagger)^{t_j}|0\rangle$$

Maintenant, nous utilisons le théorème d'expension multinomiale : $\forall k, m \in \mathbb{N}, \forall (x_1, \ldots, x_m) \in \mathbb{K}$,

$$\Big(\sum_{i=1}^{m} x_i\Big)^{k_n} = \sum_{\substack{\{k\} \\ \sum_n k_n = k}} \binom{k}{k_1, \ldots, k_m} \prod_{i=1}^{m} x_i^{k_i}$$

avec

$$\binom{k}{k_1, \ldots, k_m} = \frac{k!}{\prod_i (k_i!)}$$

.

Par conséquent, en applicant cette formule sur $x_i = U_{ji}a_i^\dagger$ et $k = t_j$:

$$(\sum_{i=1}^{m} U_{ji}a_i^\dagger)^{t_j} = \sum_{\substack{\{t_{kj}\} \\ \sum_k t_{kj} = t_j}} \binom{t_j}{t_{1j}, \ldots, t_{mj}} \prod_{i=1}^{m} (U_{ji}a_i^\dagger)^{t_{ij}}$$

Et nous obtenons :

$$|\phi\rangle = \prod_{j=1}^{m} \frac{1}{\sqrt{t_j!}} \sum_{\substack{\{t_{kj}\} \\ \sum_k t_{kj} = t_j}} \binom{t_j}{t_{1j}, \ldots, t_{mj}} \prod_{i=1}^{m} (U_{ji}a_i^\dagger)^{t_{ij}}|0\rangle$$

$$= \prod_{j=1}^{m} \sum_{\substack{\{t_{kj}\} \\ \sum_k t_{kj} = t_j}} \frac{\sqrt{t_j!}}{\prod_i t_{ij}!} \prod_{i=1}^{m} (U_{ji}a_i^\dagger)^{t_{ij}}|0\rangle$$

Maintenant,on intervertit la somme et le produit

$$\prod_{i=1}^{m}(\sum_{j=1}^{n} c_{ij}) = \sum_{f\in\mathcal{F}_{m,n}} \prod_{i=1}^{m} c_{i,f(i)}$$

avec $\mathcal{F}_{m,n} = \{f, f : [\![1,m]\!] \to [\![1,n]\!]\}$
Et on obtient :

$$|\phi\rangle = \sum_{\substack{\{t_{kj}\}\\ \sum_k t_{kj}=t_j}} \frac{\prod_{j=1}^{m}(t_j!)^{\frac{1}{2}}}{\prod_{i,j=1}^{m} t_{ij}!} \prod_{i=1}^{m}\Big(\prod_{j=1}^{m}(U_{ji}a_i^\dagger)^{t_{ij}}\Big)|0\rangle$$

Maintenant qu'on a cela, on sait que :

$$\begin{cases} \sum_i t_{ij} = t_j \text{ représente le nombre de photons à la sortie} \\ \sum_j t_{ij} = s_i \text{ représente le nombre de photons à l'entrée} \end{cases}$$

Et on peut réécrire la ligne du dessus comme ceci en appliquant les différents opérateur d'échelle :

$$|\phi\rangle = \sum_{\substack{\{t_{kj}\}\\ \sum_k t_{kj}=t_j}} \frac{\prod_{i=1}^{m}(t_i!)^{\frac{1}{2}}}{\prod_{i,j=1}^{m} t_{ij}!}\Big(\prod_{i,j}(U_{ji})^{t_{ij}}\Big)\prod_{i=1}^{m}\sqrt{(\sum_j t_{ij})!}|\sum_j t_{1j}\sum_j t_{2j}\cdots\sum_j t_{mj}\rangle$$

$$= \sum_{\substack{\{t_{kj}\}\\ \sum_k t_{kj}=t_j}} \frac{\prod_{i=1}^{m}(s_i!)^{\frac{1}{2}}\prod_{i=1}^{m}(t_i!)^{\frac{1}{2}}}{\prod_{i,j=1}^{m} t_{ij}!}\Big(\prod_{i,j}(U_{ji})^{t_{ij}}\Big)|s_1 s_2\dots s_m\rangle$$

Pour un choix de $(t_{ij})_{i,j\in\{1\dots m\}}$, on a un facteur en produit de coefficient de la matrice U à la puissance t_{ij}. Si on somme sur toutes les possibilités des $(t_{ij})_{i,j\in\{1\dots m\}}$, on obtient le permanent d'une matrice.

Pour une entrée $|s_1 s_2\dots s_m\rangle$ et une sortie $|t_1 t_2\dots t_m\rangle$, la matrice en question, qu'on appelera $U_{S,T}$ est la matrice obtenu en repétant s_j fois la $i$ème colonne et t_j fois la $j$ème ligne de la matrice U.

Pour être plus précis, et en utilisant les notations introduise par Scheel [**scheel_permanents_2004**], $U_{S,T} = U[(s_1 s_2\dots s_m),(t_1 t_2\dots t_m)] = (U_{s_i,t_j})_{i,j\in\{1\dots m\}}$.

Ainsi pour un $|\psi_{in}\rangle = |s_1 s_2\dots s_m\rangle$

$$\langle t_1 t_2\dots t_m|\psi_{out}\rangle = \langle\phi|\psi_{in}\rangle = \frac{\mathrm{Perm}(U_{S,T})}{\prod_{j=1}^{m}(s_j!)^{\frac{1}{2}}\prod_{i=1}^{m}(t_i!)^{\frac{1}{2}}}$$

et la probabilité de mesurer $|t_1 t_2\dots t_m\rangle$ avec une entrée $|s_1 s_2\dots s_m\rangle$ dans la transformation unitaire U est :

$$P_U(S,T) = |\langle t_1 t_2\dots t_m|\psi_{out}\rangle|^2 = \frac{|\mathrm{Perm}(U_{S,T})|^2}{\prod_{j=1}^{m}(s_j!)\prod_{i=1}^{m}(t_i!)}$$

Ainsi, pour une entrée S fixée, on obtient une densité de probabilité ne dépendant que de U et se calcule à l'aide d'un calcul de permanent. Dans le modèle du Boson random sampling, on fixe l'entrée : on considérera n photons qui sont chacun dans un mode différent parmis les m modes et ce seront les n premiers modes. L'état de Fock d'entrée sera $|s_1 s_2\dots s_m\rangle = |1_n\rangle = |11\dots100\dots0\rangle$

2.6 Analogie en Théorie des graphes

Le calcul du permanent peut s'avérer important en théorie des graphes.

Soit G un graphe bipartie $m\times m$. Si on appelle la fonction pm la fonction qui assigne à un graphe bipartie son nombre de match parfait. Le graphe G définie deux parties $E = \{e_1,\dots,e_m\}$ et $F = \{f_1,\dots,f_m\}$ et A la matrice d'incidence du graphe. Il y a une arrête entre $u_j \in U$ et $v_j \in V$ si et seulement si $a_{ij} = 1$. Sinon, il n'y a pas d'arrête si et seulement si $a_{ij} = 0$. Alors :

$$\mathrm{pm}(G) = \mathrm{Perm}(A)$$

La matrice A donne l'information de comment sont connectés les sommets entre les deux parties U et V et son permanent est le nombre de match parfait.Calculer le permanent d'une matrice A une matrice avec que des 0

et des 1 est une tâche P#-complet. Ainsi, cela reste une tâche très complexe, même pour une matrice plus simple.

Le Boson random samplinget ce problème de théorie de graphe sont liés en considérant l'ensemble U qui est l'ensemble des modes d'entrées du Boson samplinget V l'ensemble des modes de sorties après l'application d'une matrice unitaire U. Ainsi, le permanent de la matrice $U_{S,T}$ est la somme des poids totaux (produit des poids des arêtes) pour tous les appariements parfaits du graphe biparti. Le produit des poids des arêtes d'un match représente l'importance du match en terme de probabilité. Par conséquence plus les match sont importants, plus cette permutation sera probable par les photons.

Dans l'exemple avec $m = 2$ et $n = 2$, $b_1^\dagger b_2^\dagger|00\rangle = \sqrt{2}U_{11}U_{21}|20\rangle + (U_{11}U_{22} + U_{12}U_{21})|11\rangle + \sqrt{2}U_{12}U_{22}|02\rangle$

Le coefficient devant l'état $|20\rangle(\sqrt{2}U_{11}U_{21})$indique que pour passer d'un photon dans le mode 1 et d'un photon dans le mode 2 (état $|11\rangle$) , il faudra que le photon dans le mode 1 reste dans le mode 1 et que le photon dans le mode 2 passe dans le mode 1.

Le coefficient devant l'état $|02\rangle(\sqrt{2}U_{12}U_{22})$indique que pour passer d'un photon dans le mode 1 et d'un photon dans le mode 2 (état $|11\rangle$) , il faudra que le photon dans le mode 1 passe dans le mode 2 et que le photon dans le mode 2 reste dans le mode 2.

Le coefficient devant l'état $|11\rangle((U_{11}U_{22} + U_{12}U_{21}))$indique que pour passer d'un photon dans le mode 1 et d'un photon dans le mode 2 (état $|11\rangle$) , il y a deux possibilité :

- Soit le photon dans le mode 1 reste dans le mode 1 et le photon dans le mode 2 reste dans le mode 2.
- Soit le photon dans le mode 1 passe dans le mode 2 et le photon dans le mode 2 passe dans le mode 1.

Le calcul explicite de l'amplitude des différents états nous donnent les différentes possibilités pour qu'un état d'entré de Fock deviennent un état de sortie de Fock.

3 Complexité algorithmique

Dans ce mémoire, on s'intéressera à la complexité des algorithmes car c'est avec ces notions que nous pouvons décrire la difficulté d'un problème informatique, que ce soit en classique et en quantique.

On appelle complexité d'un algorithme son temps de calcul (autrement dit le temps nécessaire à son exécution). On s'intéresse à l'évolution de cette complexité en fonction de la taille des données ; autrement dit, étant donnée une procédure p, on s'intéresse à la variation de la durée du calcul de p(x) en fonction de la taille de l'argument x.

Cette définition de la complexité semble reposer sur une définition précise du temps de calcul ; or tout le monde sait que celui-ci, exprimé par exemple en secondes, dépend de la machine utilisée (vitesse du processeur, temps d'accès à la mémoire, etc.), et décroît rapidement avec les progrès de la technologie. Pour se rapprocher d'une notion indépendante de la technologie, on adopte la convention suivante :

Le temps de calcul de $p(x)$ est le nombre d'instructions élémentaires exécutées pendant ce calcul.

La définition d'une instruction élémentaire dépend à son tour du modèle de calcul considéré, mais un consensus est facile à trouver en pratique ; en cas de doute, on se ramène à une estimation du nombre d'instructions effectuées par le processeur. Une autre solution, employée en particulier pour démontrer les théorèmes fondamentaux de la théorie de la complexité, est d'utiliser le modèle des machines de Turing, pour lequel la définition d'une instruction élémentaire (appelée aussi transition) est non ambigüe.

Ainsi il existe plusieurs classes de complexité : $P, NP, BPP, BQP, \#P$ etc. Ces classes de complexité s'intéresse aux problème de décision et au aux problème de comptage. Pour la suite, on notera $\{0,1\}^* = \{0,1\}, \{0,1\}^2, \dots\} = \bigcup_{n\in\mathbb{N}} \{0,1\}^n$

3.1 Classes de complexité des problèmes de décisions

3.1.1 P et NP

Par commodité, on considère le plus souvent des problèmes de décision, où la tâche est de décider si une donnée $x \in \{0,1\}^*$ est dans un langage dit $L \subset \{0,1\}^*$, qui est un ensemble de chaînes de bits. Une machine qui calcule la fonction booléenne $f_L : \{0,1\}^* \longrightarrow \{0,1\}$, qui satisfait $f_L(x) = 1 \iff x \in L$, décide L. Par exemple, un langage L pourrait être donné par l'ensemble de tous les graphes pour lesquels il existe un chemin qui visite chaque sommet une fois, en codage binaire, et une chaîne $x \in L$ est le codage binaire d'une instance particulière du graphe.

Définition 1 (P). *Un langage $L \subset \{0,1\}*$ est dans la classe P s'il existe un algorithme classique $\mathcal{A}$ qui, étant donné $x \in \{0,1\}^*$ en entrée, décide si $x \in L$ en temps d'exécution polynomial en $|x|$:*

$$x \in L \iff \mathcal{A}(x) = 1$$

Cette classe est une classe de complexité très importante. Un problème est dans la classe P s'il est décidé en temps polynomial par rapport à la taille de l'entrée. On dit que le problème est décidé en temps polynomial.

Les problèmes dans P sont considérés comme « faisables », faciles à résoudre (dans le sens où on peut le faire relativement rapidement). On peut le reécrire comme ceci :

$$P = \bigcup_{c\leq 1} TIME(n^c)$$

Définition 2 (NP). *Un langage $L \subset \{0,1\}^*$ est dans la classe NP s'il existe un polynôme $p : \mathbb{N}ß\mathbb{N}$ et un algorithme classique en temps polynomial$\mathcal{V}$ (appelé le vérificateur de L) tel que pour tout $x \in \{0,1\}^*$,*

$$x \in L \iff \exists y \in \{0,1\}^{p(|x|)} : \mathcal{V}(x,y) = 1$$

.

L'abréviation NP signifie « non déterministe polynomial ».Cela ne veut pas dire que ce n'est pas polynomiale. Un problème de décision est dans NP s'il est décidé par un algorithme non déterministe en temps polynomial par rapport à la taille de l'entrée. Intuitivement, cela revient à dire qu'on peut vérifier « rapidement » (complexité polynomiale) si une solution candidate est bien solution.

P et NP sont les classes les plus importantes de la théorie de la complexité qui soulève un des problème les plus difficiles du millénaire. L'Institut de mathématiques Clay a inclus ce problème dans sa liste des sept problèmes du prix du millénaire, et offre à ce titre un million de dollars à quiconque sera en mesure de démontrer $P = NP$ ou $P \neq NP$ ou de démontrer que ce n'est pas démontrable.

Très schématiquement, il s'agit de déterminer si le fait de pouvoir vérifier rapidement une solution à un problème implique de pouvoir la trouver rapidement ; ou encore, si ce que nous pouvons trouver rapidement lorsque nous avons de la chance peut être trouvé aussi vite par un calcul intelligent.

Néanmoins, les expert du domaine de la complexité pencherai plus pour que $P \neq NP$ et nous prendrons cette information importante pour décrire le théorème d'Aaronson & et Arkhipov

3.1.2 PP ,BPP, BQP

Pour comparer la théorie de la complexité des algorithme classique versus quantique, il faut ajouter dans les définitions une part de probabilité et de hasard. Ainsi, on compare des algorithmes quantiques avec des algorithmes classiques randomisé ajoutant une part d'aléatoire.

Définition 3 (PP)**.** *On appelle PP la classe de tous les langages $L \subset \{0,1\}^*$ pour lesquels il existe un algorithme classique randomisé $\mathcal{A}$ qui décide en temps polynomial tel que pour tout $n \in \mathbb{N}$ et toutes les entrées $x \in \{0,1\}^n$*

$$x \in L \Rightarrow Pr_r[\mathcal{A}(x,r) = 1] > 1/2$$

$$x \notin L \Rightarrow Pr_r[\mathcal{A}(x,r) = 1] < 1/2$$

où la probabilité est prise sur l'aléa interne de l'algorithme.

C'est une classe de complexité probabiliste. Plus précisément c'est l'ensemble de problèmes de décision probabiliste solvable en temps polynomial avec une probabilité d'erreur inférieure à un demi.

Définition 4 (BPP (BQP))**.** *On appelle PP la classe de tous les langages $L \subset \{0,1\}^*$ pour lesquels il existe un algorithme classique randomisé $\mathcal{A}$ qui décide en temps polynomial tel que pour tout $n \in \mathbb{N}$ et toutes les entrées $x \in \{0,1\}^n$*

$$x \in L \Rightarrow Pr_r[\mathcal{A}(x,r) = 1] \geq 1/2$$

$$x \notin L \Rightarrow Pr_r[\mathcal{A}(x,r) = 1] \leq 1/2$$

où la probabilité est prise sur l'aléa interne de l'algorithme.

Les classes BPP et PP sont très proches au niveau de la définition mais a priori ne sont pas égales. En effet BPP est la classe des problèmes qui peuvent être décidés par un algorithme en temps polynomial avec une probabilité de bonne réponse supérieure à une constante elle-même strictement plus grande que $1/2$. BPP est donc incluse dans PP.

Les classes BPP et BQP sont comparable en classique et en quantique. Il est connu de $P \subset BPP$. L'autre inclusion est une question ouverte. En terme plus généraux, la question est de savoir si l'aléatoire est utile pour accélérer le calcul ou non. Il y a eu à ce sujet un changement d'avis de la part de la communauté de la complexité : jusqu'aux années 80, la plupart des chercheurs pensaient que BPP était différente de P, puis divers résultats ont bousculé cette croyance. Aujourd'hui une égalité est souvent envisagée.En plus de contenir les algorithmes qui résolvent les problème de décision en temps polynomiale, il contient par exemple les méthodes de Monte Carlo.

On peut avoir des machines plus efficaces si nécessaire, autrement dit on peut remplacer $2/3$ par $1-\varepsilon$ et $1/3$ par ε (pour tout ε petit), en ne changeant pas la classe. Ce renforcement peut être effectué en lançant plusieurs fois la machine de façon indépendante et en faisant un vote.

3.1.3 Notion de difficulté, complet et exposant de complexité

Donnons maintenant des définitions plus générale sur les complexités :

Définition 5. *Soit X une classe de complexité. On dit qu'un problème et X-difficile (ou X-hard) si on peut le ramener à un problème de complexité X par une réduction polynomiale, c'est à dire qu'il existe une fonction $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ qui permet de passer du problème difficile au problème de classe X.*

Définition 6. *Soit X une classe de complexité. On dit qu'un problème et X-complet si il est à la fois X et X-hard.*

Par exemple, le problème du voyageur est un problème NP-complet. Si on arrive à montrer que ce problème est P, alors on arrive à montrer que toute la classe NP est dans P

On peut ainsi réaliser les deux graphiques qu'on appelle diagramme d'Euler :

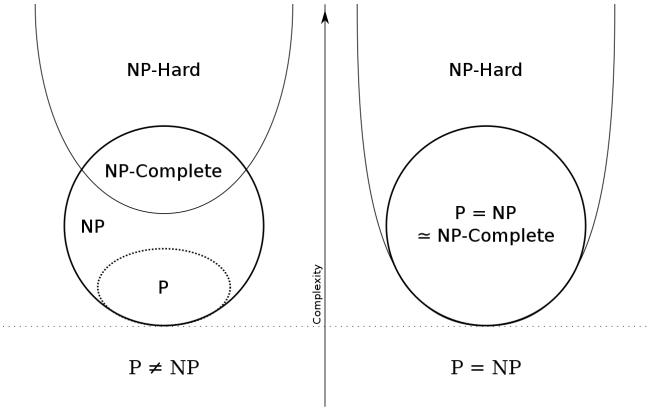

FIGURE 3 – Diagramme d'Euler

Définition 7. *Soit X et Y deux classes de complexité. On dit qu'un problème est dans la classe X^Y (nous écrivons une classe de complexité X en exposant d'une autre classe Y) si le problème est dans X ayant accès à ce que l'on peut appeler un oracle (boîte noire) ayant accès à un algorithme résolvant des problèmes arbitraires dans la classe Y .*

On peut dire que X^Y est l'ensemble des problèmes de décision dans X augmentée d'un oracle pour un problème complet de la classe Y. C'est des classes de complexité qui sont importantes en théorie dans ce que l'on va introduire comme la hiérarchie polynomial de complexité et le théorème de Toda.

3.2 Classe de complexité des problèmes de fonction et # P

3.2.1 Problème de fonction

Un problème de fonction est un problème de calcul où une seule sortie (d'une fonction totale) est attendue pour chaque entrée, mais la sortie est plus complexe que celle d'un problème de décision. Pour les problèmes de fonction, la sortie n'est pas simplement oui ou non .

Ainsi, on peut associé à chaque classe de complexité de décision un classe de complexité de fonction. Il suffit de rajouter un F devant le nom de la classe de décision analogue.

P devient FP, NP devient FNP et BPP devient $FBPP$. Tout comme P et FP sont étroitement liés, NP est étroitement lié à FNP. $FP = FNP$ si et seulement si $P = NP$.

3.2.2 Classe # P

Définition 8. *La classe de fonctions $\#P$ est la classe de toutes les fonctions $f : \{0,1\}^* \longrightarrow \mathbb{N}$ pour lesquelles il existe un algorithme classique en temps polynomial $\mathcal{A}$ et un polynôme $p : \mathbb{N} \longrightarrow \mathbb{N}$ tel que*

$$f(x) = Card\Big(\{y \in \{0,1\}^{p(|x|)} : \mathcal{A}(x,y) = 1\}\Big)$$

.

$\#P$ est la classe des fonctions qui comptent le nombre de certificats d'un problème de décision qui est dans la classe NP. La classe $\#P$ tient une place à part dans la théorie de la complexité, car ce n'est pas une classe de problèmes de décision mais une classe de fonctions de comptage de solutions.

Un problème de décision de la classe NP est souvent de la forme "Y a-t-il des instances qui satisfont certaines contraintes ?". A contrario, les problèmes de la classe $\#P$ correspondants posent la question "Combien y a-t-il d'instances qui satisfont certaines contraintes ?".

Clairement, un problème $\#P$ doit être au moins aussi dur que le problème qui lui correspond dans NP, puisque savoir combien il y a de solutions nous dit, en particulier, s'il y a au moins une solution. Ainsi, le problème de la classe $\#P$ correspondant à un problème NP-complet doit être NP-difficile. Certains problèmes de $\#P$ qui sont considérés comme difficiles correspondent à des problèmes faciles, de la classe P. Par exemple le problème du calcul du permanent est $\#P$-complet alors que le problème d'existence associé est dans P.

4 Théorème d'Aaronson & Arkhipov

Supposons que nous ayons une matrice $X \in \mathbb{C}^{nn}$, et que nous souhaitions calculer $|\text{Perm}(X)|^2$. Notez qu'il n'y a aucune perte de #P-completude ici ; c'est déjà un problème #P-complet. Naturellement, nous pourrions construire un réseau de séparateurs de faisceaux qui donne lieu à la matrice X. Cependant, il y a deux difficultés :

- X n'est pas forcément unitaire
- Avec l'échantillonage, on ne peut pas calculer exactement, mais seulement une approximation.

4.1 Encoder une matrice quelconque en matrice unitaire

Pour résoudre le problème d'unitarité de X.

Théoreme 1. *Soit $X \in \mathbb{K}^{n\times n}$ tel que $\|X\| \leq 1$. Alors il existe une matrice unitaire $U \in \mathbb{K}^{2n\times 2n}$ tel que :*

$$U = \left(\begin{array}{c|c} X & \\ \hline & \end{array}\right)$$

Preuve. Comme $\|X\| \leq 1$, on sait que la matrice $I - X^\dagger X$ est définie positive.

Ainsi, d'après la décomposition de Cholesky, il existe $Q \in \mathbb{K}^{n\times n}$ tel que $Q^\dagger Q = I - X^\dagger X$.

Posons désormais $Y = \begin{pmatrix} X \\ Q \end{pmatrix}$. Donc, $Y^\dagger Y = Q^\dagger Q + X^\dagger X = I$ et les n vecteurs colonnes forment une famille orthonormale de vecteurs dans une dimension $2n$. Par le théorème de la base incomplète, et à l'aide d'un procédé de Gram-Shmidt, nous pouvons compléter Y pour avoir une base orthornormée de $\mathbb{K}^{n\times n}$ et

$$U = \left(\begin{array}{c|c} X & B_1 \\ \hline Q & B_2 \end{array}\right)$$

□

Désormais, nous pouvons prendre n'importe quelle matrice $X \in \mathbb{K}^{n\times n}$ et la normaliser ($X \to X/\|X\|$) pour avoir $\|X\| = 1$.

On peut trouver une matrice unitaire $U = \left(\begin{array}{c|c} X & \\ \hline & \end{array}\right) \longrightarrow \langle I|\phi(U)|T\rangle = |\text{Perm}(X)|^2$

avec $I = |1, \ldots, 1, 0, \ldots, 0\rangle$

Le modèle du Boson random samplingpeut être simulée selon une probabilité dépendant de $|\text{Perm}(X)|^2$ pour n'importe quelle matrice.

4.2 Problème de l'exactitude du calcul du permanent

Ce n'est pas la même chose que d'être capable de calculer explicitement $|\text{Perm}(X)|^2$. En particulier, la permanence, et donc la probabilité, pourrait être exponentiellement petite. Nous savons que nous pouvons obtenir une estimation de $|\text{Perm}(X)|^2$ en réalisant l'expérience plusieurs fois et en calculant la moyenne de tous les résultats des mesures. Mais pour obtenir une estimation non triviale, nous devrions peut-être répéter l'expérience un nombre exponentiel de fois, auquel cas nous pourrions tout aussi bien calculer le permanent à l'aide d'un ordinateur classique.

Il faut voir maintenant s'il est difficile d'approximer le permanent d'une matrice. Nous allons démontrer ce théorème :

Théoreme 2. *Soit $X \in \mathbb{K}^{n\times n}$ tel que $\|X\| \leq 1$. Alors il existe un algorithme classique randomizé qui approxime $|\text{Perm}(X)|$ à une erreur additive $\pm\varepsilon$ en $O(\frac{n^2}{\varepsilon^2})$*

Pour prouver cela, nous avons besoin d'utiliser une autre formulation du permanent introduite par Glynn. Cette formule permet de réaliser l'algorithme classique le plus rapide connue aujourd'hui pour calculer le permanent exactement. Il est d'une complexité $O(2^n n^2)$, et peut être réduit en complexité $O(2^n n)$ en itérant les $z_1 \ldots z_n$ selon le code de Gray.

Lemme 1 (Formule de Glynn)**.** *Pour toute matrice $X \in \mathbb{K}^{n\times n}$,*

$$\text{Perm}(X) = \mathop{\mathbb{E}}_{z_1,\ldots,z_n \in \{1,-1\}} \left[z_1 \ldots z_n \prod_{i=1}^{n} (x_{i1} z_1 + \cdots + x_{in} z_n) \right]$$

En effet, quand on développe le produit, tous les termes où les termes z_i qui sont élévés au carré ou plus s'annule avec le produit $z_1 \dots z_n$. Il reste $n!$ termes qui correpondent aux termes du permanent. A partir de la formule de Glynn, on peut montrer facilement une propriété intéressante du permanent :

Théoreme 3. $|\mathrm{Perm}(X)| \leq \|X\|^n$ *et dans le cas particulier d'une matrice unitaire* U , $|\mathrm{Perm}(U)| \leq 1$

Preuve. On va utiliser la formule de Glynn mais aussi l'inégalité entre moyenne géométrique et arithmétique :

$$(\prod_{i=1}^{n} a_i)^{\frac{1}{n}} \leq \frac{1}{n} \sum_{i=1}^{n} a_i$$

□

et en utilisant la formule de Glynn :

$$\begin{aligned}\sqrt{\prod_{i=1}^{n} |x_{i1}z_1 + \cdots + x_{in}z_n|^2} &\leq \Big(\frac{\sqrt{\sum_{i=1}^{n} |x_{i1}z_1 + \cdots + x_{in}z_n|^2}}{\sqrt{n}}\Big)^n \\ &= \Big(\frac{\|Xz\|}{\|z\|}\Big)^n \\ &\leq \|X\|^n\end{aligned}$$

et nous obtenons : $|\mathrm{Perm}(X)| \leq \|X\|^n$

Maintenant que nous avons cela, nous avons un algorithme éfficace qui estime une approximation du permanent de X. Cette propriété nous permet d'assurer l'approximation de l'algorithme d'estimation du permanent. L'algoritme peut être décrit comme ci dessous :

Algorithme 1 : Algorithme d'approximation du permanent.

Résultat : Approximer le permanent avec une erreur ε
Choisir aléatoirement $\frac{1}{\varepsilon^2}$ vecteurs $Z = (z_1, \dots, z_n), \in \{1, -1\}^n$. On nommera cette ensemble V_ε ;
Calculer $\sum_{z \in V_\varepsilon} z_1 \dots z_n \prod_{i=1}^{n} (x_{i1}z_1 + \cdots + x_{in}z_n)$;
Retourner ce résultat qui correspond à une moyenne empirique.

On approxime $|\mathrm{Perm}(X)|$ avec une erreur ε en $O(\frac{n^2}{\varepsilon^2})$.

4.3 Théorème d'Aaronson

Dans cette partie, nous allons expliquer pourquoi le Boson random samplingest un problème qui pourrait montrer la suprématie quantique. Introduisons tout d'abord les complexités algorithmiques que nous aurons à traiter.

4.3.1 Compléxité du calcul du permanent

Le permanent et le déterminant sont deux valeurs que l'on peut associé à une matrice qui sont quasi semblable en terme de définition :

$$\mathrm{Perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma_i}$$

$$\det(A) = \sum_{\sigma \in S_n} \varepsilon(\sigma) \prod_{i=1}^{n} a_{i,\sigma_i}$$

Le terme multiplicatif qui change est la signature de la permutation :

$$\varepsilon(\sigma) = \prod_{1 \leq i < j \leq n} \mathrm{sgn}(\sigma(j) - \sigma(i))$$

S'il existe une formule générale de calcul du déterminant, calculer directement à partir de la formule serait une technique difficile à mettre en œuvre pour des matrices de grande taille. On lui préfère alors des méthodes de calcul plus simples comme la technique du pivot de Gauss. La méthode de Gauss, pour inverser une matrice carrée de m lignes et m colonnes (ou pour calculer son déterminant) possède une complexité $O(m^3)$, autrement

dit $O(n^{3/2})$, si $n = m2$ désigne la taille de la matrice. Donc le problème du calcul du déterminant est dans FP.

La seul chose qui change entre le déterminant et le permanent est la signature, coefficient multiplicatif devant le produit des termes de la matrice. Ces signatures $\varepsilon(\sigma) \in \{-1, 1\}$ change totalement la complexité des algorithmes de calculs.

En effet , le permanent est beaucoup plus difficile à calculer que le déterminant. Il n'existe pas d'analogue du pivot de Gauss pour les permanents. Plus précisément, le problème du calcul du permanent de matrice $0-1$ peut être vu comme un problème de comptage. En effet, il faut calculer tous les produits de termes de la matrice complexe pour chaque permutation.

En 1971, Valiant trouve la complexité du calcul du permanent :

Théoreme 4 (1971 Valiant)**.** *Le calcul du permanent du matrice* $0-1$ *est un problème* $\#P$*-complet.*

C'est à partir de ce résultat de complexité que Aaronson et Arkhipov ont vu que le lien entre le Boson random samplinget la complexité de calcul pouvait être intéressant.

Pour s'intéresser à cela, nous avons besoin d'énoncer un théorème prouver en 1983 par Stockmayer, qui nous donne l'existence d'algorithme d'approximation d'une fonction booléenne :

Théoreme 5 (1983 Stockmeyers)**.** *Soit une fonction* $f : \{0,1\}^n \to \{0,1\}$ *booléenne et*

$$p = Pr_{x\in\{0,1\}^n}[f(x) = 1] = \frac{1}{2^n} \sum_{x\in\{0,1\}^n} f(x)$$

.

Alors, $\forall c \leq 1 + 1/poly(n)$*, il existe un algorithme de classe* $FBPP^{NP^{\mathcal{O}}}$ *qui approxime* p *avec une erreur multiplicative* c

Ceux qui a poussé l'intérêt du problème Boson random samplingest ce théorème que nous expliciterons :

Théoreme 6 (Aaronson & Arkhipov (2011))**.** *Le problème exact du Boson Random Sampling n'est pas efficacement solvable par un ordinateur classique, à moins que* $P^{\#P} = BPP^{NP}$ *et que la hiérarchie polynomiale s'effondre au troisième niveau.*

Ce théorème semble compliqué mais il nous dit que si il existe un algorithme classique qui puisse simuler l'échantillonage aléatoire de Boson, alors $P = NP$. Le problème $P = NP$ est un des quelques problèmes dit du "millénaire" qui n'a toujours pas été résolu. Néanmoins, les spécialistes de la complexité conjecture que P serait différent de NP. Ainsi, sous réserve de la véracité de $P \neq NP$, il ne serait pas possible de simuler à l'aide d'un ordinateur classique le Boson random sampling.

Preuve. Soit $X \in \mathbb{R}^{n\times n}$ et un paramètre $g \in \left[1 + \frac{1}{poly(n)}, poly(n)\right]$.

D'après le Théorème de Valiant, approximer le permanent de X avec un facteur multiplicatif g est un problème $\#P$-difficile. Nous avons donc besoin de montrer approximer $|\mathrm{Perm}(X)|^2$ avec un facteur multiplicatif g est un problème aussi $\#P$-difficile.
Enfin, il faudra montrer qu'approximer $|\mathrm{Perm}(X)|^2$ est un problème $FBPP^{NP^{\mathcal{O}}}$.

D'après le théorème d'encodage des matrices dans une matrice unitaire, en posant $Y := \varepsilon X$ avec $\varepsilon = 1/\|X\| >= 2^{poly(n)}$, $\exists U \in \mathbb{K}^{2n\times 2n}$ tel que : $U = \left(\begin{array}{c|c} Y & \\ \hline Q & \end{array}\right)$

Prenons la matrice $A = \begin{pmatrix} Y \\ Q \end{pmatrix}$ qui correspond aux n premières colonnes de U. Alors, nous nous intéressons à la probabilité p_A d'avoir la sortie 1_n pour l'entrée A dans un algorithme randomisé $mathcal O$.

$$\begin{aligned} p_A &= \Pr_r[\mathcal{O}(A, r) = 1_n] \\ &= |\langle 1_n|\varphi(U)|1_n\rangle| \\ &= |\mathrm{Perm}(U_{n,n})|^2 = |\mathrm{Perm}(Y)|^2 \\ &= \varepsilon^{2n} |\mathrm{Perm}(X)|^2 \end{aligned}$$

D'après le théorème de Stockmeyer, il est possible d'approximer p_A d'un facteur multiplicatif g à l'aide d'un algorithme de classe $FBPP^{NP^{\mathcal{O}}}$.

Désormais, il faut montrer que par conséquent,$P^{\#P} = BPP^{NP}$.

On a prouvé que s'il existe un algorithme classique qui résout le problème du Boson random sampling, alors nous avons $P^{\#P} \in BPP^{NP}$. Et comme d'après le théorème de Toda, $PH \in P^{\#P}$, on a $P^{\#P} = PH = \Sigma_P^3 = BPP^{NP}$

Approximer le carrée du permanent d'une matrice est un problème $\#P$-difficile. Donc, $P^{\#P} = BPP^{NP}$, la hierarchie polynomiale s'effondre donc au troisième niveau et $P = NP$.

□

Références

[1] Dominik Hangleiter et Jens Eisert. *Computational advantage of quantum random sampling*. arXiv :2206.04079 [cond-mat, physics :quant-ph]. Nov. 2022. DOI : 10.48550/arXiv.2206.04079. URL : http://arxiv.org/abs/2206.04079 (visité le 10/11/2022).

[2] Marcel Hinsche et al. *A single T-gate makes distribution learning hard*. arXiv :2207.03140 [quant-ph, stat]. Juill. 2022. URL : http://arxiv.org/abs/2207.03140 (visité le 10/11/2022).

[3] Daniela Frauchiger, Renato Renner et Matthias Troyer. *True randomness from realistic quantum devices*. arXiv :1311.4547 [quant-ph]. Nov. 2013. URL : http://arxiv.org/abs/1311.4547 (visité le 10/11/2022).