Implémentation de l'Algorithme de Grover pour la Résolution de Systèmes Binaires Quadratiques Rapport de PFEE

Julien Mayer et Arthur Stievenard

January 15, 2024

Contenu

1 Abstract

Solving multivariate quadratic equations, known as the MQ problem, forms the foundation of various cryptosystems, particularly when instantiated over the binary field (F2) for efficiency. The prevailing approach for solving the MQ problem over F2 involves enumeration, exhibiting a time complexity of $O(2^n)$ for n variables. Grover's algorithm, anticipated to operate on large-scale quantum computers, promises a significant time reduction to $O(2^{n/2})$. The key to Grover's algorithm lies in the oracle, assessing quadratic equations at a superposition of potential inputs. Our work based on the ressearch paper "Solving binary MQ with Grover's algorithm" by Peter Schwabe and Bas Westerbaan introduces two distinct quantum circuits serving as oracles. Remarkably, our findings indicate that a relatively modest quantum computer with just 92 logical qubits can compromise MQ instances designed for 80-bit pre-quantum security.

Furthermore, our collaboration took place within the dynamic research environment of the Laboratoire LIP6 at Sorbonne Université. This institution, founded in 1997 and affiliated with Sorbonne Université, stands as a leading research center in computer science. With a focus on diverse domains, including artificial intelligence, theoretical computer science, distributed systems, cybersecurity, databases, and modeling, LIP6 fosters innovation and interdisciplinary collaboration.

In summary, our project, conducted under the guidance of Ludovic Perret, Associate Professor at LIP6, delves into the implementation of Grover's algorithm for solving binary quadratic systems. Through meticulous exploration and experimentation, we contribute to the ongoing efforts to harness the potential of quantum computing in cryptographic problem-solving.

Keywords: Grover's algorithm, multivariate quadratics, quantum resource estimates.

2 Présentation de la société

Le Laboratoire LIP6, affilié à Sorbonne Université, incarne un centre de recherche dynamique spécialisé en informatique. Fondé en 1997, il résulte de la fusion des forces de recherche de l'Université Pierre et Marie Curie, désormais intégrée à Sorbonne Université.

2.1 Domaines de Recherche

Le LIP6 consacre son expertise à la recherche en informatique, explorant un éventail de domaines comprenant l'intelligence artificielle, l'informatique théorique, les réseaux et systèmes distribués, la sécurité informatique, les bases de données, ainsi que la modélisation et la simulation. La recherche est réalisée au sein de 22 équipes (dont 3 communes avec Inria Paris) articulées autour de quatre axes transverses :

- Intelligence artificielle et sciences des données
- Architecture, systèmes et réseaux
- Sécurité, sûreté et fiabilité
- Théorie et outils mathématiques pour l'informatique

2.2 Organisation

La structure du laboratoire s'articule autour d'équipes de recherche spécialisées, chacune se concentrant sur des sujets spécifiques. La synergie entre ces équipes favorise la collaboration interdisciplinaire et externe, stimulant ainsi l'innovation et l'avancement des connaissances.

2.3 Historique

L'Institut Blaise-Pascal du CNRS est fondé en 1946, avec Joseph Pérès comme directeur. Elle devient en 1994 une fédération des laboratoires d'informatique du campus Jussieu (Fédération d'Unités FU0007 du CNRS), regroupant alors LAFORIA, LITP, MASI. En 1997, différents laboratoires fusionnent et forment le Laboratoire d'Informatique de Paris 6 (LIP6). Le LIP6 a évolué au fil des années pour devenir un pilier incontournable de la recherche en informatique en France. Les fondateurs ont impulsé une vision inspirante qui continue d'animer les chercheurs et les étudiants engagés dans ses activités.

2.4 Personnel et Collaborations

Aujourd'hui sous la direction de Fabrice Kordon, le laboratoire contient aujourd'hui environ 550 collaborateurs rassemblant une diversité de talents, allant des chercheurs aux enseignants-chercheurs, des doctorants aux ingénieurs de recherche.

- Enseignants-chercheurs : 160
- Chercheurs : 28
- Personnels d'appui à la recherche : 23
- Doctorants : 189
- Post-doctorants : 54

Il entretient des collaborations actives avec d'autres laboratoires de recherche, des institutions académiques, des entreprises et des organismes gouvernementaux.

2.5 Infrastructure et Projets Notables

Grâce à leur infrastructure technologique avancée, le LIP6 participe activement à des projets de recherche notables, tant nationaux qu'internationaux. Il se distingue par la régularité de ses publications dans des revues et des conférences scientifiques prestigieuses.

2.6 Engagement Académique

Le laboratoire s'investit dans la communauté académique en organisant des conférences, des séminaires et en prenant part à des événements scientifiques. Cette contribution active favorise la diffusion des connaissances et nourrit le dialogue au sein de la communauté. Le LIP6 se consacre à la modélisation et la résolution de problèmes fondamentaux, ainsi qu'à la mise en œuvre et la validation des solutions au travers de partenariats académiques et industriels.
Le Laboratoire LIP6 à Sorbonne Université joue un rôle pivot dans la recherche en informatique, mariant expertise, collaboration et engagement académique pour repousser constamment les limites de la connaissance dans ce domaine passionnant. Cette institution se profile ainsi comme un bastion de l'innovation et un acteur essentiel dans l'écosystème de la recherche informatique en France et au-delà.

3 L'organisation du PFEE

3.1 Les interlocuteurs côté client

Ludovic Perret :

Actuellement Associate Professor au Laboratoire d'Informatique de Paris 6, occupant cette fonction depuis février 2007. Parallèlement à son rôle universitaire, il enseigne à Epita les technologies quantiques et la cryptographie post-quantique.
En outre, Ludovic Perret exerce la fonction de mentor pour les start-ups au sein du programme MyStartup de Sorbonne University depuis mars 2023. Il est également le co-fondateur de CryptoNext Security depuis juin 2019, une entreprise opérant dans la région de Paris, France, où il occupe le poste de premier PDG.

Ludovic Perret a également joué un rôle central en tant que point de contact privilégié pour le Projet de Fin d'Études (PFEE). En qualité d'interlocuteur exclusif, il a assumé la responsabilité de guider et de superviser le travail des étudiants, offrant ainsi un encadrement précieux tout au long du projet. Son implication s'est manifestée par la fourniture d'un sujet de recherche, matérialisé sous la forme d'un document de recherche. Ce sujet a constitué la base sur laquelle les étudiants ont construit leurs travaux de fin d'études.

3.2 Les étudiants et de leur rôle

Nous sommes Julien Mayer et Arthur, étudiants en 5e année à EPITA, spécialisés dans la majeure Quantique. Fortement passionnés par l'informatique, les nouvelles technologies et la physique, le choix de la majeure Quantique s'est avéré être l'option idéale pour notre parcours académique. En tant que duo collaboratif, nous avons eu l'opportunité de travailler étroitement sous la direction de Ludovic, le responsable du projet.
Notre implication s'est articulée en deux phases distinctes. Durant la première, nous avons exploré de concert le fonctionnement du premier oracle en suivant les directives claires fournies par Ludovic. Lors de la deuxième phase du projet, nos rôles ont évolué. Julien a pris en charge l'implémentation du deuxième oracle, cherchant à concrétiser les concepts acquis. En parallèle, Arthur s'est consacré aux tests visant à valider le fonctionnement du premier oracle étudié précédemment. Cette répartition stratégique des tâches a maximisé notre efficacité, assurant une progression fluide dans le développement du projet.
Notre collaboration étroite, notre complémentarité et notre dévouement ont joué un rôle essentiel dans le succès de cette entreprise. En partageant une passion commune et en démontrant un engagement fort dans le domaine de la majeure Quantique, nous contribuons activement à l'avancement significatif du projet.

3.3 La gestion de projet

La réalisation de notre Projet de Fin d'Études (PFEE) a impliqué une gestion de projet rigoureuse, basée sur des outils collaboratifs modernes et des réunions régulières. Notre approche a été centrée sur une communication transparente, des objectifs clairs, et une collaboration étroite entre les membres de l'équipe.

Réunions et Communication :
Nous avons organisé des réunions régulières, soit en ligne ou en présentiel, alternant entre les locaux d'EPITA et le site de l'université de Sorbonne. Ces rencontres se déroulaient à des intervalles hebdomadaires ou bimensuels, offrant ainsi des opportunités pour discuter de l'avancement du projet. Durant ces réunions, nous fixions des sous-objectifs, examinions les solutions proposées, ajustions nos objectifs en fonction des besoins du projet, et répartissions efficacement les tâches entre moi-même et Julien, mon collègue.

Outils de Collaboration :
Nous avons utilisé plusieurs outils de collaboration pour faciliter le travail à distance. Dropbox a été choisi comme espace de stockage principal pour partager les documents, les ressources et les captures d'écran. Les réunions en ligne ont été organisées via Zoom, fournissant une plateforme stable pour les discussions synchrones, la présentation de notre travail, et la clarification des doutes. GitHub a été notre plateforme de gestion de versions, où le code a été partagé, collaboré, et suivi au fil du temps.

Suivi du Code :
Un dépôt GitHub dédié a été créé, permettant à moi-même et à Julien de travailler simultanément sur le code. Cette approche collaborative nous a permis de suivre les modifications, de gérer les conflits de fusion, et d'assurer une cohérence dans le développement du projet. Chaque semaine, le code était envoyé à notre maître de stage pour une évaluation régulière. Cela garantissait une supervision constante de notre progression et permettait d'ajuster la direction du projet en fonction des retours reçus.

Interactions et Collaboration :
Notre équipe, composée de Julien, moi-même, et notre maître de stage, a maintenu une communication ouverte et régulière. Les interactions étaient principalement centrées sur le trio, évitant toute complexité liée à une équipe plus étendue. Cela a favorisé une prise de décision rapide et des ajustements agiles en fonction des exigences du projet.

En résumé, la gestion de projet mise en place a permis une coordination efficace entre les membres de l'équipe, une communication transparente, et un suivi précis de l'évolution du code. Ces pratiques ont été essentielles pour garantir le succès de notre Projet de Fin d'Études.

4 Le Planning récapitulatif

4.1 Compréhension du sujet

Le Projet de Fin d'Études (PFEE) avait pour objectif principal l'implémentation de l'algorithme de Grover, tel que décrit dans le papier de recherche intitulé "Solving binary MQ with Grover's algorithm" rédigé par Peter Schwabe et Bas Westerbaan de la Digital Security Group à l'Université Radboud. Plus spécifiquement, le projet visait à appliquer l'algorithme de Grover pour résoudre des systèmes binaires quadratiques.

Pour des tailles de problèmes couramment utilisées en cryptographie, l'algorithme classique le plus performant connu est l'énumération en code Gray. Dans l'article, Bo-Yin Yang, Jiun-Ming Chen, and Nicolas T. Courtois estime que des algorithmes asymptotiquement plus rapides ne deviennent avantageux que pour des systèmes avec environ n = 200 variables. Cependant, sur un ordinateur quantique, on peut utiliser l'algorithme de Grover. Pour appliquer l'algorithme de Grover, il est nécessaire de fournir un oracle approprié : un circuit quantique qui vérifie si un vecteur (xi) est une solution pour un système donné. Tout circuit booléen peut être traduit en un circuit quantique équivalent, cependant, les traductions naïves nécessitent généralement une grande quantité de registres auxiliaires.

Problème : Un système d'équations quadratiques sur F_2 sous une forme pratique est donné par un 'cube' $\lambda_{ij}^{(k)}$ pour i, j, k dans F_2, où $\lambda_{ij}^{(k)} = 0$ dès que $i > j$. L'objectif est de trouver $x_1, \ldots, x_n \in F_2$ tels que

$$\sum_{1 \le i \le j \le n} \lambda_{ij}^{(1)} x_i x_j = 1, \ldots, \sum_{1 \le i \le j \le n} \lambda_{ij}^{(m)} x_i x_j = 1.$$

Nous avons travaillé sur l'implémentations de deux oracles distinctes, toutes deux présentées dans l'article de référence. La première approche a privilégié l'utilisation d'un nombre plus élevé de qubits, tout en cherchant à minimiser la profondeur de l'algorithme. En revanche, la seconde implémentation a opté pour une approche inverse, en utilisant moins de qubits mais en augmentant la profondeur de l'algorithme.

Le premier oracle que nous construisons utilisera au plus $n + m + 2$ registres de qubits et $O(mn^2)$ temps pour un système de m équations quadratiques sous une forme pratique avec n variables. Notre deuxième oracle n'utilisera que $n + 3$ registres de qubits, mais nécessitera approximativement le double du temps. Il est important de noter que le système contient également des termes linéaires tels que $x_i^2 = x_i$.

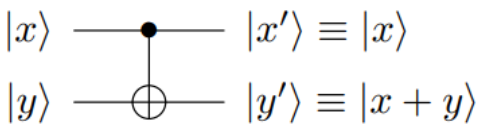
Exemple. Nous utiliserons le petit système suivant :

$$x_1(1 + x_2 + x_3) + x_2 x_3 = 1$$

$$x_2(1 + x_3) = 1$$

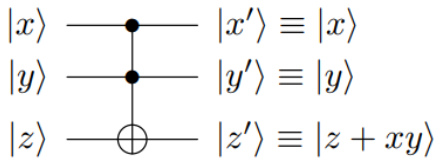
Voici les portes quantiques que nous utiliserons pour les oracles présentés au dessus. Toutes les portes quantiques que nous utiliserons sont les homologues quantiques des portes classiques réversibles.

Porte 1 : Nous utiliserons une porte CNOT (controlled not - également appelée porte Feynman) pour calculer le XOR. En tant qu'unitaire, elle est définie sur la base computationnelle par $\mathrm{CNOT}|x\rangle|y\rangle = |x\rangle|x+y\rangle$



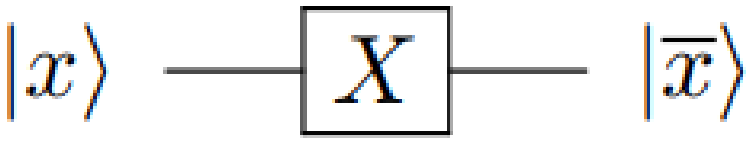
x	y	x'	y'
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Porte 2 : Pour calculer le AND, nous utiliserons la porte Toffoli T. En tant qu'unitaire, elle est définie par $\mathrm{T}|x\rangle|y\rangle|z\rangle = |x\rangle|y\rangle|z+xy\rangle$

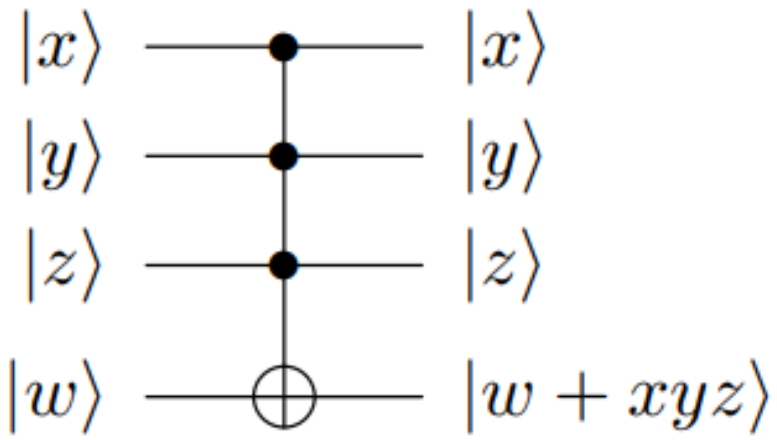


x	y	z	x'	y'	z'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

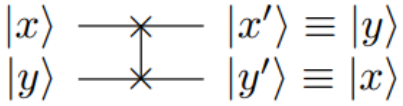
Porte 3 : Pour calculer le NOT, nous utilisons la porte X. En tant qu'unitaire, elle est définie par $\mathrm{X}|x\rangle = |x\rangle = |1+x\rangle$



Porte 4 : Pour calculer le AND de plusieurs bits, nous utiliserons la porte Toffoli à n qubits (Tn). Il s'agit d'une porte controlled-not avec n-1 bits de contrôle. Son action en tant qu'unitaire sur la base computationnelle est $\mathrm{Tn}|x_1\rangle \dots |x_n\rangle = |x_1\rangle \dots |x_{n-1}\rangle |x_n + (x_1 \dots x_{n-1})\rangle$.



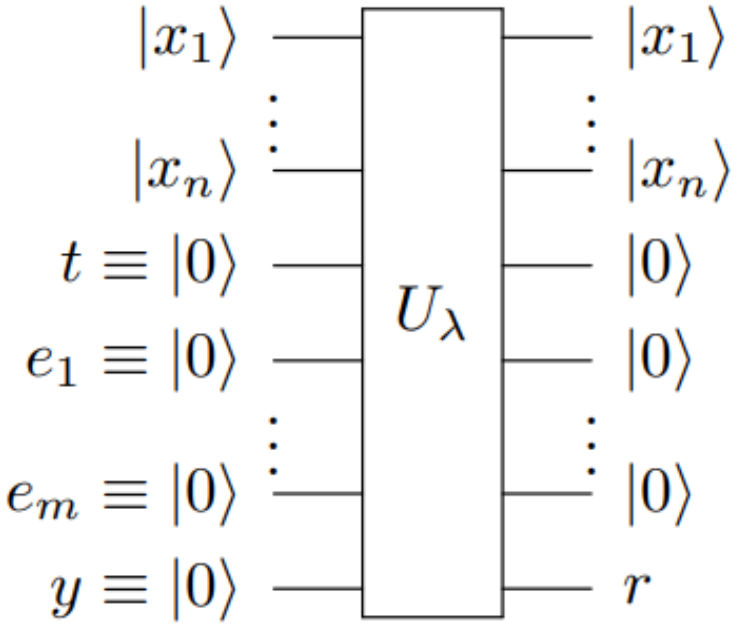
Porte 5 : Pour le deuxième oracle, nous voulons échanger des bits, ce qui est fait avec la porte d'échange à 2 qubits (SWAP). En tant qu'unitaire, elle est définie par $\mathrm{SWAP}|x\rangle|y\rangle = |y\rangle|x\rangle$.



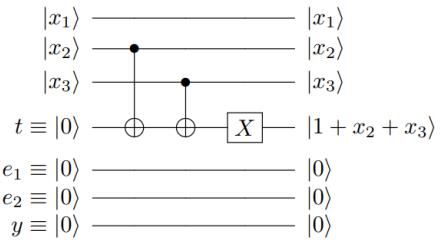
x	y	x'	y'
0	0	0	0
0	1	1	0
1	0	0	1
1	1	1	1

Il est prévu que les portes X, SWAP et CNOT seront peu coûteuses à exécuter et à corriger les erreurs sur un ordinateur quantique, tandis que les portes Toffoli (à n qubits) seront coûteuses.

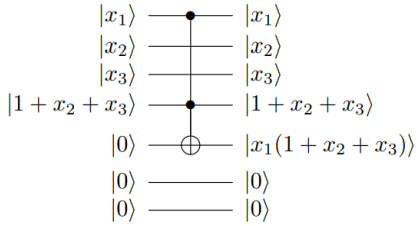
Premier Oracle :
Le premier oracle, conforme à l'algorithme de Grover pour MQ sur F_2, est construit à l'aide d'un circuit U_λ. Ce circuit, vérifiant si (x_i) est une solution d'un système de m équations quadratiques en n variables dans une forme pratique, utilise au plus $n + m + 2$ registres de qubits et nécessite $O(mn^2)$ temps pour un système de m équations quadratiques sous une forme pratique avec n variables.
Le circuit agit comme suit, où le registre finale $y = |1\rangle$ si (x_i) est une solution et $|0\rangle$ sinon.



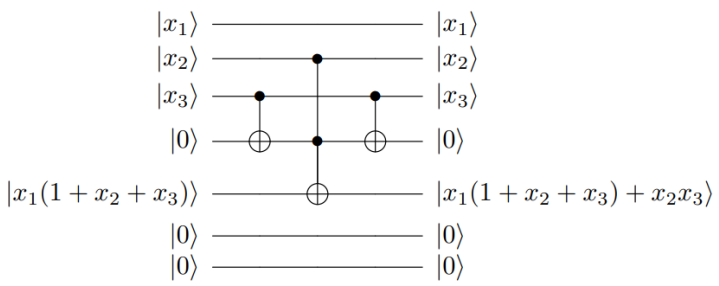
Les premiers n registres sont les entrées et doivent être initialisés avec $x_1, \ldots, x_n$. Le circuit ne les modifie pas, même temporairement. Le registre suivant est un registre auxiliaire étiqueté t, initialement à $|0\rangle$. Les m registres suivants, étiquetés $e_1, \ldots, e_m$, sont tous initialisés à $|0\rangle$. Le dernier registre est un registre de sortie étiqueté y.
Le circuit complet est construit étape par étape, utilisant les portes vu précédement telles que CNOT, Toffoli, et X pour reconstruire le système équations par équations. Chaques équations du système est assemblé dans le registre temporaire 't' avant d'être stocké dans son registre 'e' dédié.



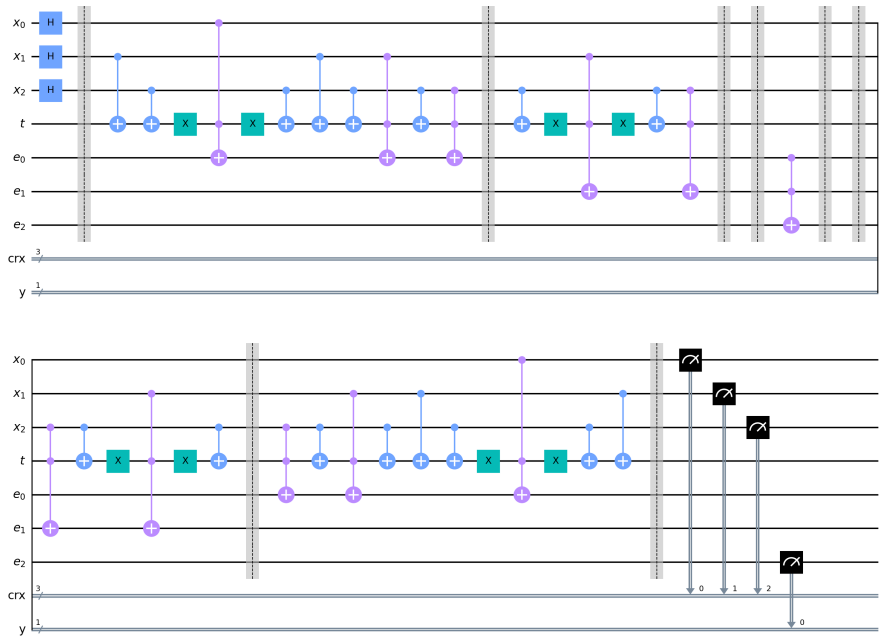
Using one Toffoli gate, we put $x_1 y_1^{(1)}$ into e_1. In our example:



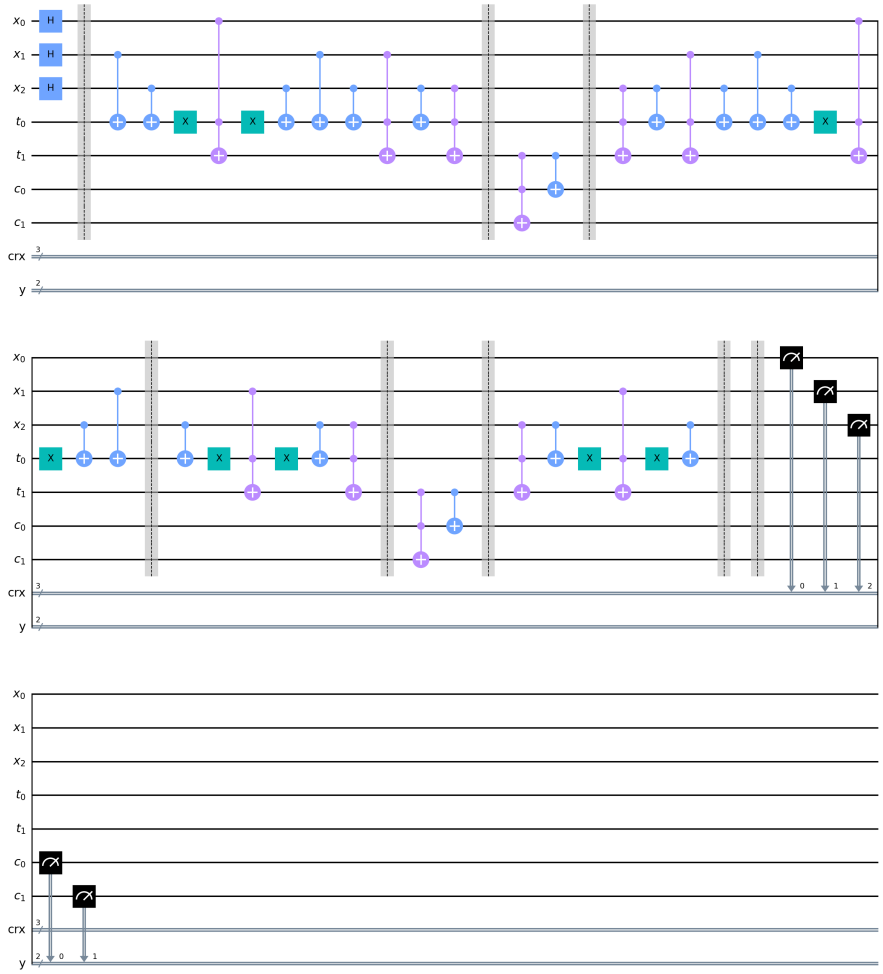
On applique ensuite le circuit inverse sur les registre x_n et t pour les remettre dans les conditions initiales avant de traiter l'équation suivante. On peut aussi simplifier le circuit a ce moment en utilisant la fonction transpile de qiskit de manière a supprimer les portes inutile du circuit.



Un fois que toutes les equations du systèmes sont représenté dans les registres 'e', on applique une potre de Toffoli avec tout les registre 'e' en controle sur le registre 'y'. L'objectif final est de vérifier si le système d'équations est satisfait et d'obtenir le résultat dans le registre y. Si toutes les équation sont satisfaites pour une entrée x_n, le résultat dans le registre 'y' sera $|1\rangle$.



Deuxième Oracle :
Le deuxième oracle, plus complexe que le premier, nécessite moins de qubits mais environ le double du nombre de portes. Dans cet oracle, plutôt que de stocker chaque équation satisfaite individuellement, on compte simplement le nombre d'équations satisfaites. Au lieu des m registres séparés, seulement $\lceil \log_2 m \rceil$ registres sont nécessaires pour agir comme un compteur. À la fin, l'oracle vérifie si la valeur dans le compteur est égale à m, ce qui est fait avec des portes X judicieusement placées et une Toffoli multi-qubits.
L'incrément du compteur nécessite une structure spécifique pour éviter l'inefficacité de l'encodage binaire standard. Pour ce faire, un circuit est construit en utilisant un polynôme primitif sur F_2, correspondant à la multiplication par x dans le champ $F_2[x]/(x^3+x+1)$ dans l'exemple donné. Ce circuit, qui peut être généralisé pour n'importe quel nombre de qubits, est construit en choisissant un polynôme primitif de degré approprié.



La table suivante compare le nombre maximum de chaque type de porte utilisée dans le deuxième oracle par rapport au premier, en prenant l'exemple d'un système de 81 équations en 85 variables.

Type de Porte	Qubits	X	CNOT	Toffoli
Premier Oracle	168	27,710	1,156,680	13,770
Deuxième Oracle	94	55,250	2,316,276	27,702

Pour trouver une solution à ce système d'exemple, l'oracle sera exécuté environ 2^{40} fois, entrelacé avec des réflexions, totalisant ainsi environ 2^{61} portes exécutées lors de l'utilisation du deuxième oracle.
Cette dualité dans les approches d'implémentation a permis d'explorer les compromis entre la complexité matérielle et la complexité temporelle de l'algorithme de Grover dans le contexte spécifique de la résolution de systèmes binaires quadratiques. En atteignant cet objectif, le PFEE a contribué à la compréhension pratique et à l'expérimentation des variations de l'algorithme de Grover dans des applications concrètes.

Le rôle central de Ludovic Perret en tant que superviseur et évaluateur a été crucial pour orienter les efforts de l'équipe, en s'assurant que les implémentations étaient conformes aux spécifications de l'article de recherche et en fournissant des conseils stratégiques pour surmonter les défis techniques rencontrés. Le PFEE a ainsi représenté une opportunité unique pour les étudiants de contribuer à la mise en œuvre concrète d'un algorithme quantique novateur dans le domaine de la résolution de systèmes binaires quadratiques.

4.2 L'architecture technique

L'architecture technique du projet a été construite autour de Python3, avec l'utilisation du package Qiskit pour la réalisation des circuits quantiques. Cette approche a permis de bénéficier de la puissance de Python dans le développement rapide et la simplicité du code, tout en exploitant les fonctionnalités de Qiskit pour la conception et la simulation de circuits quantiques.De plus, une grande partie des logiciels liés au domaine quantique se développe en Python, facilitant ainsi le choix.

Initialement envisagée en Julia pour l'écriture des équations quadratiques, l'équipe a finalement opté pour une approche entièrement basée sur Python, en raison de sa familiarité et de son haut niveau d'abstraction. Cette décision a simplifié le processus de développement en capitalisant sur la compétence existante des membres de l'équipe, permettant ainsi une mise en œuvre plus efficace de l'algorithme de Grover dans le contexte spécifique des systèmes binaires quadratiques.

Un autre choix à faire résidait dans la traduction des équations en code. Le papier de référence suggérait une approche basée sur des tenseurs, c'est-à-dire des matrices pour chaque équation. Ces matrices étaient supposées être triangulaires, soit supérieures, soit inférieures, pour éviter les doublons. C'est ce qui se faisait de standard, donc c'est cette decision qui à ete prise. Et elle a été essentielle pour garantir la simplicité de l'interaction de l'utilisateur avec le système.

En résumé, l'architecture technique du projet a été minutieusement pensée pour maximiser l'efficacité du développement, la collaboration et la compréhension globale du projet, tout en s'adaptant aux compétences et aux préférences des membres de l'équipe.

4.3 Problèmes Rencontrés

4.3.1 Problème avec le Compteur du Deuxième Oracle

Une difficulté majeure sur le plan technique a émergé lors de la mise en œuvre du compteur pour le deuxième oracle. Le document de référence suggérait une approche non conventionnelle pour le comptage en évitant l'encodage binaire standard. Au lieu de cela, il proposait un circuit de compteur basé sur un polynôme primitif sur F_2, démontrant son efficacité avec un exemple concret de circuit à 3 qubits.
Ce choix introduisait une complexité supplémentaire dans la mise en œuvre du compteur, car l'incrément standard de ce compteur était difficile à réaliser efficacement sans l'utilisation de registres auxiliaires. Toutefois, après une évaluation approfondie, nous avons opté pour l'encodage binaire standard, jugé plus compréhensible et, après une révision du professeur, nous avons constaté que cette méthode n'entraînait pas de changement significatif de complexité en termes de nombre de qubits. Cette approche a été jugée plus pertinente pour notre projet.

4.3.2 Incohérence entre le Nombre de Portes Indiqué et les Résultats du Deuxième Oracle

Un autre défi a émergé lors de la comparaison entre le nombre de portes indiqué dans l'énoncé pour l'ordre de grandeur et celui que nous avons obtenu pour le deuxième oracle. Nous avons suspecté que cela pourrait être lié au compteur, mais malgré cette suspicion, les résultats obtenus étaient corrects.
Cette divergence entre les indications théoriques et les résultats pratiques a soulevé des questions sur la cohérence des informations fournies dans l'énoncé du problème. Cependant, il est important de noter que, malgré ces divergences apparentes, les résultats finaux obtenus étaient conformes aux attentes, ce qui souligne la complexité et la nuance associées à la mise en œuvre pratique d'algorithmes quantiques.

5 Conclusion

5.1 Le résultat final

L'implémentation de Grover que nous avons développée peut modéliser un système d'équations binaires quadratiques sous la forme d'une liste de matrices symétriques. Ces matrices représentent les coefficients quadratiques dans les équations, et chaque équation est modélisée comme une combinaison linéaire de termes quadratiques.
Considérons un exemple concret de système d'équations binaires quadratiques modélisé par deux matrices symétriques :

$$\text{Matrice 1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Cette matrice correspond à l'équation $x_1 + x_1 \cdot x_2$.

$$\text{Matrice 2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Cette matrice correspond à l'équation $x_0 \cdot x_2 + x_1$.

En appliquant notre implémentation de Grover avec ces matrices, nous obtenons les mêmes solutions pour les deux oracles, confirmant la cohérence des résultats. Les solutions obtenues sont
Solution oracle 1 : ['011', '110', '101']
Solution oracle 2 : ['011', '110', '101']
Ces résultats peuvent être vérifiés à l'aide de tests spécifiques, où les combinaisons de bits générées par les solutions correspondent aux solutions du système d'équations quadratiques. En fournissant des matrices symétriques en entrée, nous avons démontré que notre implémentation de Grover peut efficacement résoudre des systèmes d'équations binaires quadratiques et produire des résultats cohérents à travers deux oracles.

5.2 Les apprentissages acquis

Expérience Professionnelle et Travail d'Équipe
La réalisation de ce Projet de Fin d'Études (PFEE) a fourni une expérience professionnelle enrichissante. Travailler en équipe, a renforcé nos compétences en communication, en collaboration, et en gestion de projet. Les réunions régulières, la répartition des tâches, et la résolution de problèmes en tandem ont contribué à une dynamique de travail efficace.

Approfondissement des Connaissances en Grover
Le projet a permis d'approfondir considérablement notre compréhension de l'algorithme de Grover au-delà de ce qui avait été enseigné en cours. En travaillant sur la résolution concrète de problèmes mathématiques, nous avons exploré des nuances et des applications pratiques de Grover.
Maitrise Accentuée de Qiskit
L'implémentation des circuits quantiques avec Qiskit a été une étape cruciale du projet, offrant une maîtrise approfondie de cette bibliothèque. La manipulation de qubits, la conception de circuits, et la gestion des simulations ont renforcé nos compétences pratiques dans le domaine de l'informatique quantique.
Découverte du Langage Julia
Au cours du projet, nous avons découvert et exploré le langage de programmation Julia. Bien que son intégration complète pour la rédaction des équations n'ait pas été réalisée, l'expérience a introduit une nouvelle dimension dans notre palette de compétences, ouvrant la voie à une exploration plus approfondie dans des projets futurs.

5.3 Les points a améliorer

5.3.1 Intégration de Julia pour la Rédaction des Équations

Nous avons envisagé d'écrire les équations en Julia en raison de ses capacités mathématiques avancées. L'objectif était de traduire ensuite ces équations en Python pour les besoins d'implémentation dans Qiskit. Bien que cette approche aurait facilité les tests et vérifications car Julia est pratique pour la résolution d'équations, sa mise en œuvre s'est avérée complexe et, par manque de temps, n'a pas été pleinement réalisée.

5.3.2 Utilisation de Git pour le Partage de Code

Le partage de code via Dropbox, bien qu'adapté à notre projet actuel, peut devenir trop lourd dans des projets plus vastes et prolongés. Une alternative plus efficace serait l'utilisation de Git uniquement, une solution de contrôle de version, offrant une gestion plus robuste, des fonctionnalités de suivi des modifications et une meilleure collaboration entre les membres de l'équipe. Cette approche permettrait une gestion plus optimale du code source sur le long terme.

5.4 Les éléments et connaissances utiles

Maîtrise Préalable de Julia : Une connaissance préalable plus approfondie du langage Julia aurait été bénéfique pour faciliter l'intégration de la rédaction des équations en Julia, comme initialement envisagé. Une formation préalable sur ce langage aurait potentiellement accéléré la mise en œuvre.