Échantillonnage et comptage avec les algorithmes variationnels quantiques

par

Julien Drapeau

Mémoire présenté au département de physique
en vue de l'obtention du grade de maître ès science (M.Sc.)

FACULTÉ des SCIENCES
UNIVERSITÉ de SHERBROOKE

Sherbrooke, Québec, Canada, 3 mars 2025

Le _____

le jury a accepté le mémoire de M. Julien Drapeau dans sa version finale.

Membres du jury

Professeur Stefanos Kourtis
Directeur de recherche
Département de physique

Professeur André-Marie Tremblay
Membre interne
Département de physique

Professeur Baptiste Royer
Président rapporteur
Département de physique

« Je vis parce que les montagnes ne savent pas rire, ni les vers de terre chanter. »

- Emil Cioran

Sommaire

En exploitant à la fois les ressources classiques et quantiques, les algorithmes variationnels quantiques peuvent exécuter des tâches computationnelles complexes sur des ordinateurs quantiques bruités, sans nécessiter de correction d'erreurs. Ces algorithmes visent notamment à résoudre des problèmes d'optimisation combinatoire, se plaçant ainsi en concurrence avec des algorithmes classiques perfectionnés au fil des décennies. Une approche complémentaire consiste à se tourner vers des problèmes intrinsèquement plus complexes : les problèmes de comptage. Ces derniers, qui consistent à déterminer le nombre de solutions aux problèmes de décision, pourraient se révéler plus accessibles aux algorithmes variationnels quantiques qu'aux algorithmes classiques.

Dans le cadre de ce mémoire, un algorithme surnommé VQCount est introduit pour la résolution de problèmes de comptage. VQCount s'appuie sur l'équivalence entre l'échantillonnage aléatoire et le comptage approximatif pour estimer le nombre de solutions à un facteur multiplicatif près, en exploitant l'ansatz quantique à opérateurs alternants comme générateur de solutions. Cette approche nécessite un nombre d'échantillons polynomial selon la taille du problème même lorsque ce dernier possède un nombre exponentiel de solutions, une amélioration exponentielle par rapport à des travaux précédents.

À l'aide de simulations de réseaux de tenseurs, la performance de VQCount avec des circuits de faible profondeur est étudiée sur des instances synthétiques de deux problèmes #P-difficile, positif #1-in-3SAT et positif #NAE3SAT, en employant comme générateurs de solutions l'algorithme quantique d'optimisation approximative ainsi que l'ansatz quantique à opérateurs alternants avec forçage de Grover. Un compromis entre la probabilité de succès et l'uniformité de l'échantillonnage du générateur est observé et exploité pour atteindre un gain en efficacité exponentiel par rapport à l'échantillonnage par rejet naïf. Ces résultats obtenus soulignent le potentiel et les limitations des algorithmes variationnels quantiques pour le comptage approximatif.

Mots-clés : Algorithmes variationnels quantiques, Échantillonnage aléatoire, Comptage approximatif, Auto-réductibilité, Satisfaisabilité, Réseaux de tenseurs

Remerciements

Table des matières

Liste des figures

Introduction

En tirant parti du parallélisme, de la superposition et de l'intrication, les algorithmes quantiques remettent en question les limites classiques du calcul, promettant des accélérations exponentielles pour certains types de problèmes. L'algorithme de Shor permet par exemple de factoriser des entiers naturels en offrant une accélération superpolynomiale par rapport aux meilleurs algorithmes classiques connus [1]. L'utilité actuelle du calcul quantique demeure toutefois incertaine pour plusieurs raisons. Les algorithmes quantiques ne s'appliquent présentement qu'à une classe de problèmes restreints, possédant un éventail d'applications limitées. De plus, les ordinateurs quantiques du moment, nommés ordinateurs quantiques à échelle intermédiaire bruités, sont affligés par différentes complications rendant tout calcul excessivement difficile : un nombre limité de qubits, une connectivité limitée et la présence d'erreurs limitant la taille du circuit.

Pour contourner ces difficultés, les algorithmes variationnels quantiques (« Variational Quantum Algorithms ») (VQA) fût conçus pour exploiter les mécanismes de l'algorithmie quantique tout en prenant avantage de la puissance du calcul classique [2]. Ceux-ci reposent sur des circuits quantiques paramétrés dont les paramètres sont optimisé par un optimiseur classique pour effectuer des tâches computationnelles complexes. Étant basée sur l'optimisation de paramètres, cette stratégie est en mesure de considérer toutes les difficultés précédentes d'un coup tout en limitant la taille des circuits quantiques. De nombreux travaux au cours des dernières années ont tenté d'améliorer les VQA en introduisant des stratégies pour mieux exploiter la structure des problèmes. Malgré ces études, il est présentement inconnu si ces algorithmes possèdent un avantage par rapport aux algorithmes classiques.

Les VQA sont typiquement appliqués à des problèmes d'optimisation combinatoire, comme le problème de coupe maximale et d'ensembles indépendants. Comme des solveurs classiques pour ce type de problème ont été développés au fil de décennies, il est peu évident de montrer la supériorité des VQA. Le projet de ce mémoire offre une perspective différente sur cette approche. Plutôt que d'entrer en compétition avec des algorithmes classiques déjà performants, les algorithmes variationnels quantique sont appliqués à la résolution de problèmes vastement plus complexes : les problèmes de comptage. Ceux-ci, éléments de la classe complexité #P, se place plus haut que les problèmes d'optimisation dans la hiérarchie de la complexité computationnelle.

Pour ce faire, l'équivalence entre le comptage approximatif et l'échantillonnage quasi uniforme, montré par Jerrum, Valiant et Vazirani [3], est utilisée. Cette correspondance donne lieu un algorithme randomisé de comptage approximatif, surnommé algorithme de JVV, permettant l'estimation à un facteur multiplicatif près du nombre de solutions à un problème de comptage auto-réduction par l'échantillonnage d'une distribution aléatoire quasi uniforme. Cet algorithme se base sur la propriété d'auto-réductibilité de certains problèmes, c'est- à-dire la possibilité d'utiliser la structure inhérente au problème pour résoudre celui-ci. La distribution préparée par un VQA peut alors être utilisée comme pour la génération de solutions à cet algorithme, résolvant de cette façon le problème de comptage.

L'algorithme VQCount, introduit dans ce mémoire, permet de faire le pont entre les VQA et les problèmes de comptage. VQCount prépare une distribution d'états contenant une superposition de solutions au problème de comptage avec une haute probabilité en utilisant l'ansatz quantique à opérateurs alternés avec forçage de Grover [4]. L'algorithme de comptage approximatif est alors employé pour obtenir un estimé du nombre de solutions. Toutefois, pour prendre avantage de la propriété d'auto-réductibilité, une procédure d'auto-réduction, modifiant le circuit quantique, est utilisée dans l'algorithme VQCount.

Grâce à des simulations de réseaux de tenseurs, cette étude évalue la performance de VQCount sur des circuits quantiques de faible profondeur appliqués à des instances synthétiques de deux problèmes #P-difficiles : positif #1-in-3SAT et positif #NAE3SAT. Les solutions sont générées à l'aide de l'algorithme quantique

d'optimisation approximative (QAOA) [5] et de l'ansatz quantique à opérateurs alternants avec forçage de Grover (GM-QAOA). L'analyse met en évidence un équilibre entre la probabilité de succès et l'uniformité de l'échantillonnage, qui permet d'obtenir un gain exponentiel en efficacité par rapport à une approche naïve par rejet. Ces résultats révèlent à la fois les atouts et les limites des algorithmes variationnels quantiques pour le comptage approximatif.

Le chapitre 1 décrit en profondeur les problèmes de comptage avec le langage de la théorie de la complexité. L'algorithme de JVV et ses concepts sous-jacents, c'est-à-dire l'auto-réductibilité, l'échantillonnage aléatoire et le comptage approximatif, sont présentés au chapitre 2. Les algorithmes variationnels quantiques sont présentés de manière générale au chapitre 3 avec un focus sur QAOA et GM-QAOA. L'algorithme VQCount, résultat de ce travail, est détaillé au chapitre 4. La performance de l'algorithme VQCount est finalement caractérisé au chapitre 5. L'annexe A montre que la procédure d'auto-réduction est bien applicable au, alors que l'annexe B introduit les méthodes de réseaux de tenseurs utilisé pour la simulation de circuit quantique de l'algorithme VQCount.

Le travail effectué au cours de ce projet a mené à la publication d'un manuscrit sur *arXiv* : *Rajouter le lien*. Ce mémoire prend ainsi son inspiration de cet écrit.

Chapitre 1

Complexité du comptage

Qu'est-ce qu'un problème de comptage ? Simplement, un tel problème consiste à déterminer le nombre de solutions respectant un ensemble de contraintes. Les problèmes de comptage font partie des problèmes computationnellement difficiles, ce qui signifie qu'il est peu probable que ces problèmes soient résolubles efficacement. Les problèmes de comptage surgissent dans de nombreuses disciplines, avec des applications en raisonnement probabiliste [6-8], en fiabilité des réseaux [9, 10], en mécanique statistique [11] et en intelligence artificielle [12]. Contrairement à certains problèmes étudiés en algorithmie quantique, comme l'échantillonnage de bosons [13] ou l'échantillonnage de circuits aléatoires [14], une solution à ces problèmes peuvent avoir un impact concret, justifiant alors l'étude de tels problèmes.

Ce chapitre tente de décrire le cadre théorique entourant la classe des problèmes de comptage en détails. Pour ce faire, celle-ci est définie à l'aide du langage de la théorie de la complexité à la section 1.1. Le problème de comptage prototypique, le problème de satisfaisabilité, sert d'exemple et est décrit à la section 1.2. Celui-ci est d'une grande importance pour ce projet et est ainsi référencé tout au long de ce mémoire. La section 1.3 explique l'importance des méthodes approximatives alors que la section 1.4 décrit les solveurs modernes pour les problèmes de comptage. Finalement, le comportement des instances aléatoires pour ces problèmes est présenté à la section 1.5, où des transitions de phase font étonnamment leurs apparitions.

Il est souvent favorable de décrire la théorie de la complexité à l'aide du concept de *machine de Turing*. Cependant, ce mémoire suppose que la thèse de Church-Turing

soit vraie, et donc que n'importe quel modèle de calcul puisse être utilisé de manière équivalente, pour faire abstraction de ce langage et simplifier les idées présentées.

1.1 Classes de complexité

La théorie de la complexité s'intéresse à la classification des problèmes algorithmiques en *classes de complexité*, c'est-à-dire en ensembles de problèmes de même complexité. Classifier un problème permet de caractériser les ressources nécessaires pour sa résolution par un algorithme. Les problèmes d'une même classe possèdent une difficulté inhérente similaire, ce qui permet le choix d'un algorithme et de ressources appropriées en conséquence. Savoir qu'un problème n'est pas réalistement résoluble, ou plus précisément intraitable, limite les attentes. Sachant ceci, la recherche dans cette direction peut s'avérer grandement utile. La théorie de la complexité cherche aussi à comparer les problèmes de différentes complexités. Ces comparaisons permettent de comprendre l'espace des problèmes en plus grande profondeur. Par exemple, il est évident que certains problèmes sont plus facile à résoudre que d'autres. Comparer des problèmes faciles avec des problèmes difficiles peut aider à comprendre ce qui rend un problème difficile et donc à trouver des algorithmes résolvant efficacement les problèmes plus complexes. Des liens, nommés réductions, peuvent aussi être définis au sein d'une même classe d'une complexité. Un algorithme efficace pour un problème pourrait aussi être efficace pour un problème similaire s'il existe une réduction entre ceux-ci. Les classes de complexité, de manière similaire au modèle de la machine de Turing, tentent de définir de manière abstraite la difficulté d'un problème. Peu importe le matériel informatique à notre disposition, un problème d'une classe donnée ne devrait pas changer de classe. Un problème trivial devrait rester trivial peu importe la quantité de ressources utilisée.

Comment est-il possible de déterminer la complexité d'un problème ? Pour ce faire, les classes de complexité se basent sur les ressources indispensables à la résolution du problème : le temps et la mémoire. Afin de trouver la solution à un problème, un programme doit effectuer un certain nombre d'opérations, limité dans le temps par le matériel informatique. On parle alors de *complexité en temps*. Afin

de produire un résultat final, le programme doit garder en mémoire les résultats intermédiaires. Ceux-ci doivent être sauvegardés dans le matériel informatique afin d'être réutilisés ultérieurement. Comme la quantité d'information conservée est aussi un facteur limitant pour le matériel informatique, on parle donc de *complexité en espace*.

La complexité en temps et en espace d'un problème est définie selon la taille de celui-ci. Afin de capturer cette dépendance, on cherche à trouver une loi d'échelle encapsulant la difficulté d'un problème en fonction de sa taille. Le temps et la mémoire quantifient bien les ressources nécessaires des algorithmes. Par contre, ceux-ci dépendent du matériel informatique utilisé. Il est attendu qu'un ordinateur moderne soit bien plus performant qu'une des premières machines analogues. Comment retirer cette dépendance de la notion de complexité ? Pour ce faire, on fait appel à la notation asymptotique, communément appelé la *notation* $\mathcal{O}$. La notation asymptotique caractérise la vitesse de croissance d'une fonction en ne considérant que son comportement global à l'infini. Les coefficients ainsi que les termes asymptotiquement inférieurs ne sont pas considérés. Par exemple, pour une taille de problème n, la résolution de celui-ci pourrait demander un temps exponentiel $\mathcal{O}(2^n)$ et une mémoire polynomiale $\mathcal{O}(n)$. Remarquons qu'il n'y a aucune dépendance au matériel informatique : deux ordinateurs différents doivent effectuer le même nombre d'opérations et sauvegarder la même quantité d'information. Un de ces ordinateurs pourrait toutefois résoudre le problème plus rapidement si celui-ci peut effectuer un plus grand nombre d'opérations par seconde ou accéder plus rapidement à sa mémoire. L'attrait des classes de complexité vient donc en partie de cette abstraction du matériel informatique.

La quantification de ces ressources permet la séparation de plusieurs problèmes : il est en effet souhaitable d'être capable de séparer les algorithmes efficaces de ceux qui ne le sont pas. Commençons par définir deux classes de complexité particulièrement importantes : P et NP. Pour ce faire, un certain type de problème doit d'abord être défini. Un *problème de décision* regroupe simplement tous les problèmes pouvant se répondre par oui ou non. Pour tout problème de décision A, on peut représenter celui-ci par une fonction $A(x) \in \{ 0, 1 \}$, où x représente une instance du problème. Les problèmes de décision se manifestent fréquemment, tant en informatique qu'en physique. Ceux-ci se présentent sous diverses formes : Est-ce qu'un nombre x est

premier ? La configurapportn x représente-t-elle un état fondamental du système donné ? Est-ce qu'il existe un chemin x parcourant une seule fois toutes les villes d'une région en parcourant au maximum une distance d ?

Quand peut-on dire qu'un problème de décision est résoluble efficacement ? La classe de complexité P, pour « temps polynomial », tente de répondre à cette question. Informellement, un problème de la classe P est un problème de décision qui peut être résolu en temps polynomial. Un problème est donc considéré comme efficacement résoluble ou *traitable*, s'il appartient à la classe P.

Définition 1.1 – Classe de complexité P

Une fonction A fait partie de la classe de complexité P si et seulement si elle prend la forme

$$A(x) = \exists y$$

et calculable en temps polynomial, c'est-à-dire en temps $O(n^c)$ pour une taille $n = |x|$ et une constante c, où $|y| = \text{poly}(|x|)$.

Soit, par exemple, le problème de décision du test de primalité A. Ce problème cherche à déterminer si un entier naturel x est premier ou composé. Ce problème peut être résolu, c'est-à-dire qu'il est possible de calculer $A(x) = \exists y$, en temps polynomial $\tilde{O}(\log(n)^{12})$ [15], où la notation $\tilde{O}$ signifie que les termes polylogarithmiques sont aussi cachés.

La relation entre un calcul en temps polynomial et un calcul efficace semble évidente de prime abord. La thèse de Cobham–Edmonds indique en effet qu'un problème algorithmique peut être résolu efficacement s'il est résoluble en temps polynomial [16, 17]. Cependant, certains problèmes ne possèdent pas de solutions efficaces en pratique. Par exemple, un problème peut appartenir à la classe P, mais être doté d'un grand coefficient limitant le calcul. Cela n'étant pas le cas pour la majorité des problèmes, cette supposition s'avère malgré tout une bonne règle empirique.

Une deuxième classe particulièrement importante en théorie de la complexité est la classe NP, pour « temps polynomial non déterministe ». Celle-ci regroupe les problèmes de décision dont les solutions sont vérifiables en temps polynomial.

Généralement, ces problèmes sont formulés sous la forme suivante : Existe-t-il une solution, vérifiable en temps polynomial, au problème donné ? Malgré que les solutions sont vérifiables efficacement, déterminer l'existence d'une telle solution n'est pas nécessairement possible en temps polynomial. Une métaphore souvent utilisée pour la description d'un problème NP est celle d'une aiguille dans une botte de foin. Trouver cette aiguille parmi la quantité énorme de brins de foin est un défi de taille. Par contre, une fois l'aiguille trouvée, il n'y a aucun doute qu'il s'agit bien d'une aiguille.

Définition 1.2 – Classe de complexité NP

Une fonction A fait partie de la classe de complexité NP si et seulement si elle prend la forme

$$A(x) = \exists y \mid B(x, y)$$

pour une fonction $B \in \mathsf{P}$, où $|y| = \text{poly}(|x|)$.

On appelle B le vérificateur du problème de décision A et y le certificat ou le témoin pour l'entrée x. Un exemple de problème NP est le problème du commis voyageur. Soit un graphe x représentant les villes d'une région particulière. Est-ce que le commis voyageur peut visiter toutes les villes exactement une fois et revenir à son point de départ en parcourant une distance inférieure à d ? Dans ce cas, la fonction A détermine le chemin parcouru alors que la fonction B détermine si un chemin y est de taille inférieure à d.

La classe NP est aussi définie comme la classe de problèmes pouvant être résolu par une machine de Turing non déterministe en temps polynomiale, d'où son nom. Cette définition est toutefois équivalente à la définition précédente, c'est-à-dire la classe de problèmes vérifiable par une machine de Turing déterministe en temps polynomial [18]. Notons que la classe P est contenue dans la classe NP. En effet, comme un problème de P est résoluble en temps polynomial, alors nécessairement une solution à ce problème peut aussi être vérifiée en temps polynomial. Cependant, une question, faisant partie des problèmes du prix du millénaire [19], demeure ouverte : est-ce que $\mathsf{P} = \mathsf{NP}$? La conjecture largement répandue répond à cette question par la négative. L'hypothèse de temps exponentiel [20] suggère par exemple que certains problèmes de NP sont résolubles en temps exponentiel, un résultat

significatif sur la difficulté de ces problèmes. En conséquence, un problème est *intraitable* s'il n'est pas résoluble en temps polynomial, par opposition à un problème traitable.

La classe de complexité d'intérêt dans le cadre de ce mémoire est la classe #P introduit par Valiant [21]. Cette classe, définie par extension à la classe NP, cherche non seulement à déterminer si un problème de décision possède une solution, mais aussi à spécifier le nombre de solutions à ce problème. Ainsi, un *problème de comptage* consiste à trouver le nombre de solutions d'un problème de décision. Un problème de comptage est donc défini par rapport à un problème de décision. Prenons par exemple le problème du commis voyageur. Le problème de décision généralement défini cherche un trajet de taille inférieure à une certaine distance. Le problème de comptage adjoint exige alors le nombre de trajets possibles de taille inférieure à la distance donnée.

Définition 1.3 – Classe de complexité #P

Une fonction A fait partie de la classe de complexité #P si et seulement si elle prend la forme

$$A(x) = |\{\, y \mid B(x,y) \,\}|$$

pour une fonction $B \in \mathsf{P}$, où $|y| = \text{poly}(|x|)$ pour toutes les valeurs y prises par $B(x,y)$.

Par définition, un problème de la classe #P est au moins aussi difficile qu'un problème de la classe NP. Connaître le nombre de solutions à un problème de décision indique effectivement s'il existe au moins une solution à un problème de décision. Cependant, les problèmes de comptage intéressants possèdent fréquemment un nombre exponentiel de solutions, impliquant une plus grande complexité inhérente.

Comme que les problèmes de la classe P font aussi partie de la classe NP, et que les problèmes de cette dernière appartiennent aussi à la classe #P, il est nécessaire de définir un concept supplémentaire spécifiant la difficulté d'un problème au sein d'une classe. Intuitivement, un problème est *difficile* pour une classe de complexité s'il est au moins aussi difficile que tous les problèmes de la classe. Plus formellement, un problème difficile pour une classe donnée signifie qu'il existe une *réduction*, c'est-à-dire une transformation en temps polynomial d'un problème à un autre, entre ce

problème et tous les autres problèmes de la classe. De plus, un problème est *complet* s'il est difficile et aussi membre de la classe. La figure 1.1 éclaircit ces concepts. Les problèmes complets d'une classe représentent alors les problèmes les plus difficiles de celle-ci. Par exemple, le problème du commis voyage est dit NP-complet, car il fait partie des problèmes considérés comme difficile de la classe NP. Il est toutefois crû que celui-ci ne fasse pas partie de la classe P selon la conjecture $\mathsf{P} \neq \mathsf{NP}$.

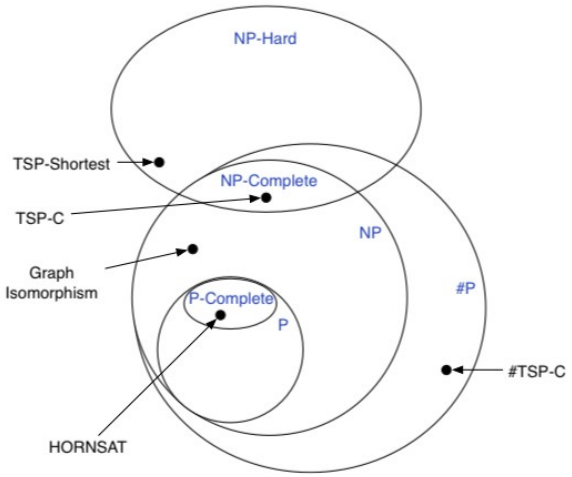

FIGURE 1.1 – *Refaire la figure.*

L'importance des problèmes #P au sein de la théorie de la complexité s'observe entre autres par le théorème de Toda [22]. Ce résultat marquant montre que tous les problèmes de la hiérarchie polynomiale PH peuvent être résolus en temps polynomial avec un oracle résolvant instantanément les problèmes #P. La hiérarchie polynomiale généralise les classes de complexité P, NP et co-NP en capturant les classes de problèmes exprimés par une alternance de quantificateurs d'existence ($\exists$) ou d'universalité ($\forall$) (voir la définition 1.2 pour un exemple contenant un seul quantificateur d'existence). Ainsi, cette hiérarchie PH est comprise dans la classe de complexité $\mathsf{P}^{\#\mathsf{P}}$, c'est-à-dire les problèmes résolubles avec un oracle #P. Comme la hiérarchie polynomiale contient de nombreux problèmes importants, ce théorème suggère l'incroyable puissance des problèmes de comptage.

Théorème 1.1 – Théorème de Toda

La hiérarchie polynomiale PH est contenue dans $\mathsf{P}^{\#\mathsf{P}}$.

L'ordinateur quantique bouleverse le domaine de la complexité classique. Les problèmes autrefois intraitables deviennent potentiellement résolubles efficacement à l'aide d'algorithmes quantiques. Deux nouvelles classes de complexité font alors leurs apparitions pour décrire les problèmes résolubles à l'aide de matériel informatique quantique. La classe BQP généralise la classe BPP, pour « temps polynomial probabiliste à erreur bornée », c'est-à-dire la classe de problèmes résoluble avec une probabilité d'erreur inférieure à $1/3$, pour les ordinateurs quantiques. De plus, la classe de complexité QMA, pour « Merlin Arthur Quantique » se définit par rapport à la classe BQP de manière analogue à la classe NP pour la classe P. La théorie de la complexité quantique étudie les classes de complexité quantiques dans l'objectif de déterminer les problèmes où les algorithmes quantiques comportent un avantage par rapport aux algorithmes classiques. Ce mémoire avance la recherche dans cette direction en étudiant la performance des algorithmes quantiques en comparaison avec les algorithmes classiques.

1.2 Satisfaisabilité booléenne

Le problème de *satisfaisabilité booléenne* ou problème SAT est particulièrement important dans la théorie de la complexité. Montré comme NP-complet par le théorème de Cook-Levin [23, 24], il fut à la base de la définition de NP-complétude et du problème $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$. Celui-ci est aussi couramment utilisé dans la preuve de réductions de problèmes au sein de la classe de complexité NP. Le problème SAT a une multitude d'applications, comme le diagnostic des fautes d'un circuit logique ou la planification en intelligence artificielle [25], en partie grâce à la facilité de formuler ces applications à l'aide de formules propositionnelles.

Une *formule propositionnelle*, ou une expression booléenne, est un ensemble de variables booléennes, $x_i \in \{\texttt{FAUX}, \texttt{VRAI}\}$, reliées par des opérateurs booléens de conjonctions (OU, $\vee$), de disjonctions (ET, $\wedge$) ainsi que de négation (NON, $\neg$). Un *littéral* désigne dans ce contexte une variable booléenne ou sa négation. Par exemple, l'expression $(x_1 \wedge x_2) \vee \neg x_3$ est une formule booléenne composée des variables x_1, x_2 et x_3, ainsi que des littéraux x_1, x_2 et $\neg x_3$. Notons que l'équivalence $\texttt{FAUX} \leftrightarrow 0$ et $\texttt{VRAI} \leftrightarrow 1$ est utilisée dans ce mémoire par commodité.

Un problème SAT se décrit par une formule propositionnelle. Résoudre le problème consiste à déterminer s'il existe une combinaison de variables qui rend la formule logiquement vraie, c'est-à-dire tel que l'évaluation de celle-ci donne 1. Une telle formule est alors dite satisfaisable.

Définition 1.4 – Problème SAT

Soit une constante $n \geq 1$ et une formule propositionnelle $\varphi(x_1, x_2, \ldots, x_n)$ où $x_i \in \{0, 1\}$. Existe-t-il une assignation des variables $x_1, x_2, \ldots, x_n$ telle que φ soit satisfaisable, c'est-à-dire que $\varphi(x_1, x_2, \ldots, x_n) = 1$?

Dans l'étude du problème SAT, les formules propositionnelles sont souvent exprimées en *forme normale conjonctive* (« Conjunctive Normal Form ») (CNF). On parle alors de formules CNF. Celles-ci consistent en une conjonction d'une ou de plusieurs *clauses*, où une clause est une disjonction d'un ou plusieurs littéraux. Cela implique que toute clause doit contenir au moins un littéral évaluant à 1 pour que la formule soit satisfaisable. Toute formule propositionnelle peut être réécrite en forme normale conjonctive en utilisant les lois de l'algèbre booléenne.

Exemple 1.1 – Problème SAT

La formule CNF

$$\varphi(x_1, x_2, x_3) = (x_1 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3)$$

est satisfaisable car $\varphi(1, 0, 0) = (1 \vee 0) \wedge (\neg 1 \vee 0 \vee \neg 0) = 1$. Au contraire, la formule CNF

$$\varphi(x_1) = (x_1) \wedge (\neg x_1)$$

n'est pas satisfaisable car $\varphi(x_1) = 0$ peu importe le choix de x_1.

Le problème kSAT constitue un cas spécial du problème SAT, où le nombre de littéraux appartenant à chaque clause d'une formule CNF est restreint à k littéraux au maximum. Notons que le problème kSAT est trivial pour $k = 1$, résoluble en temps linéaire pour $k = 2$ [26], et NP-complet pour $k \geq 3$ [27]. Surprenamment, les problèmes de comptage correspondant, #2SAT et #3SAT, appartiennent tous les deux à la classe #P-complet [9]. Ainsi, compter le nombre de solutions à un

problème NP-complet peut être difficile même s'il est possible de trouver une solution efficacement.

Dans ce mémoire, deux variantes de 3SAT seront portées à l'étude : le problème Pas-Tous-Égaux 3SAT (« Not-All-Equal 3-Satisfiability ») (NAE3SAT) et le problème 1-dans-3 3SAT (« One-in-Three 3-Sastisfiability ») (1-in-3SAT). Comme le problème SAT, ces problèmes appartiennent aussi à la classe de complexité NP-complet. Les versions monotones de ces problèmes, où la négation de variables n'est pas permise, sont étonnamment aussi NP-complet par le théorème de dichotomie de Schaefer [28]. Ces problèmes de décision peuvent être associés aux problèmes de comptage #NAE3SAT et #1-in-3SAT de la classe de complexité #P.

Définition 1.5 – Problème NAE3SAT

Soit une formule CNF $\varphi(x_1, x_2, \ldots, x_n)$ pour laquelle chaque clause C contient au maximum 3 littéraux. Existe-t-il une assignation des variables $x_1, x_2, \ldots, x_n$ telle que φ soit satisfaisable tout en s'assurant que tous les littéraux de chaque clause C ne soient pas égaux ?

Notons qu'un problème NAE3SAT peut être décrit par un problème 3SAT. Soit une formule CNF f exprimant un problème 3SAT. Une formule CNF g, représentant le problème NAE3SAT associé à f, est construite en transformant chaque clause $(x \vee y \vee z)$ de f en $(x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$. Cette nouvelle contrainte renforce alors la condition supplémentaire, c'est-à-dire que les littéraux de chaque clause ne peuvent être tous égaux. Cette relation illustre bien une symétrie cachée derrière le problème NAE3SAT ; chacune des variables possèdent en moyenne la même probabilité d'être vraie ou fausse.

Définition 1.6 – Problème 1-in-3SAT

Soit une formule CNF $\varphi(x_1, x_2, \ldots, x_n)$ pour laquelle chaque clause C contient au maximum 3 littéraux. Existe-t-il une assignation des variables $x_1, x_2, \ldots, x_n$ telle que φ soit satisfaisable tout en s'assurant qu'exactement un littéral de chaque clause C soit logiquement vrai ?

De la même façon que pour le problème NAE3SAT, le problème 1-in-3SAT se transforme en un problème 3SAT en transformant chaque clause $(x \vee y \vee z)$ en

$(x \vee y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee \neg z) \wedge (\neg x \vee \neg y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$, de manière à encoder la contrainte additionnelle du problème 1-in-3SAT.

1.3 Intraitabilité, approximation et optimisation

Les problèmes algorithmiques des classes de complexité NP-difficile et #P-difficile sont considérés comme intraitables en raison de leur complexité intrinsèque. En effet, il est improbable qu'un algorithme puisse résoudre exactement ces problèmes en temps polynomial. Par exemple, bien que certaines instances du problème SAT contenant plus d'un million de variables sont résolubles efficacement, d'autres instances de moins d'un millier de variables ne peuvent être résolus par les solveurs de pointe [29]. Cette difficulté s'accentue pour le problème #SAT, où même une centaine de variables peut s'avérer trop complexe.

L'intraitabilité de tels problèmes mène alors à un paradigme différent ; Sachant qu'il est irréaliste de résoudre certains problèmes exactement, est-ce qu'il est possible de trouver une solution approximative de manière efficace ? L'exactitude des solutions est alors sacrifié pour une performance accrue.

Les problèmes de décision ne possédant que deux solutions, oui ou non, il n'est pas possible de fournir une solution approximative. Un *problème d'optimisation*, défini en conjonction à un problème de décision, est alors une notion plus adéquate pour incorporer la notion d'approximation. Ces problèmes ne cherchent plus à trouver la solution optimale, mais plutôt à obtenir une réponse suffisamment près de la valeur optimale. Un exemple de problème d'optimisation, particulièrement étudié dans le domaine de l'optimisation quantique en raison de sa simplicité d'implémentation sur du matériel informatique quantique, est le problème de coupe maximum (« Maximum Cut ») (Max-Cut). Ce problème cherche une coupe séparant les sommets d'un graphe en deux ensembles complémentaires tel que le nombre d'arêtes séparant les deux ensemble soit maximal. Le problème Max-Cut étant NP-complet, il est difficile d'obtenir une coupe optimale, mais une coupe sous-optimale peut possiblement être trouvée efficacement.

Un problème d'optimisation Π est constitué d'un ensemble d'instances valides I_Π, où chaque instance $x \in I_\Pi$ possède un ensemble de solutions faisables $S_\Pi(x)$. Une fonction objectif obj_Π, aussi nommée fonction de coût ou de perte, quantifie la qualité d'une solution approximative y de x en lui assignant un nombre réel. En conséquence, résoudre approximativement un problème d'optimisation correspond à minimiser ou maximiser la fonction de coût. Une mesure de succès fréquemment utilisée, autant classiquement que quantiquement, est le *rapport d'approximation*,

$$\alpha(x, y) = \frac{\text{obj}_\Pi(x, y)}{\text{OBJ}_\Pi(x)}, \tag{1.1}$$

où $\text{OBJ}_\Pi(x) = \min_y \text{obj}_\Pi(x, y)$. Ce type de problème est formalisé sous le nom de problème d'optimisation NP, mais une définition détaillée est évitée ici par simplicité. Bien que les algorithmes d'approximation heuristiques offrent fréquemment de bons résultats, ceux-ci ne possèdent aucune garantie quant à la qualité de ces solutions.

Pour pallier ce problème, des algorithmes approximatifs avec garantis sont définis en fonction de deux paramètres : la tolérance ε et la confiance δ. La tolérance indique l'erreur multiplicative maximale de l'approximation et la confiance indique la probabilité de succès de l'algorithme. Ce couple de paramètres est souvent utilisés pour décrire les algorithmes approximatifs.

Un *algorithme d'approximation de tolérance* ε pallie ce problème en garantissant une solution optimale à une erreur multiplicative près [30]. Pour un problème de minimisation Π, cet algorithme f produit une solution s pour toutes instances $x \in I_\Pi$ tel que $f_\Pi(x, y) \leq \varepsilon(|x|) \cdot \text{OBJ}(x)$ où $|x|$ est la taille de l'instance et $\varepsilon \geq 1$. Cette définition peut être détendue en permettant à l'algorithme de produire une telle solution avec une certaine probabilité. Cette variante, l'*algorithme d'approximation randomisé de tolérance* ε, introduit des concepts pertinents pour le chapitre 2.

Théorème 1.2 – Algorithme d'approximation randomisé

Un algorithme d'approximation randomisé de tolérance ε pour un problème de minimisation Π est un algorithme aléatoire prenant en entrée une instance $x \in D_\Pi$ et une tolérance ε et qui produit une solution $y \in S_\Pi(x)$ tel que

$$\Pr[f_\Pi(x, y) \leq \varepsilon(|x|) \cdot \mathrm{OBJ}(x)] \geq \frac{1}{2}$$

en temps polynomial.

Ces algorithmes se généralisent facilement pour un problème de maximisation. En relaxant la condition d'exactitude, il est attendu qu'un gain en possible en terme d'efficacité. Cependant, cela n'est pas toujours aussi clair. Il existe en effet certains problèmes où trouver une solution approximative en haut d'un certain rapport d'approximation demeure intraitable tel le problème de couverture par ensembles [31]. Les algorithmes d'approximation s'appliquent aussi aux problèmes de comptage, tel que présenté à la section 2.3.

1.4 Comptage de modèles

Sachant désormais que le comptage est un problème difficile, comment résoudre celui-ci ? Le comptage des solutions d'une formule propositionnelle est spécifiquement connu dans la littérature sous le nom de *comptage de modèles*. De nombreuses méthodes, autant exactes qu'approximatives, ont été développées pour sa résolution [32].

Les méthodes exactes attaquent généralement le problème en explorant exhaustivement l'espace des solutions possibles, de manière similaire à l'algorithme de Davis-Putnam-Logemann-Loveland (DPLL) pour les problèmes SAT [33]. Les algorithmes de recherche locale pour SAT, tel l'algorithme WalkSAT, peuvent aussi être étendus pour résoudre #SAT. Toutefois, les solveurs les plus performants actuellement sont basés sur les réseaux de tenseurs, un object mathématique provenant du domaine de la matière condensée [34-36]. Ces réseaux sont utiles dans de nombreux autres domaines et sont d'ailleurs utilisés dans ce travail pour la simulation de

circuit quantique et font donc l'objet de l'annexe B. Bien que différents développements accentuent la taille des systèmes résolubles exactement, les solveurs ont de la difficulté à trouver l'ensemble de solutions de l'espace de recherche.

Les méthodes approximatives allègent ce problème à l'aide d'heuristiques fournissant des estimations avec ou sans garanties. Plusieurs applications du comptage ne nécessitent pas un résultat exact ; distinguer la différence entre 10^{30} et $10^{30}+1$ solutions n'est pas toujours pertinent. L'algorithme de Stockmeyer, qui approxime le nombre de solutions à un facteur deux avec un nombre polynomial d'appels à un oracle NP en s'appuyant sur les fonctions de hachage, fut un pas majeur pour le comptage approximatif [37]. Plusieurs modifications ont propulsé le domaine vers des performances accrues de sorte que la majorité des méthodes modernes se basent effectivement sur les fonctions de hachage. Cependant, une méthode alternative due à Jerrum, Valiant et Vazirani utilise plutôt la relation entre l'échantillonnage aléatoire de solutions et le comptage approximatif pour résoudre le problème. Malgré l'originalité de cette idée, la recherche dans cette direction s'est estompée en raison de la difficulté de l'échantillonnage de solutions avec les conditions nécessaires. Ce travail relance cette possibilité en pourvoyant une nouvelle façon d'échantillonner avec les algorithmes variationnels quantiques. Une discussion approfondie est présentée au chapitre 2.

Les avancées en physique, tel les réseaux de tenseurs, présentent des avantages pour le comptage de modèle, mais l'inverse est aussi vrai. Le problème du comptage est intimement reliée au domaine de la mécanique statistique. En effet, déterminer la fonction de partition d'un système à température nulle est en réalité équivalent à un problème de comptage [38]. Considérons la fonction de partition Z pour un Hamiltonien H où l'énergie fondamentale est fixée à $E_0 = 0$,

$$Z = \sum_i e^{-\beta E_i}\,, \tag{1.2}$$

où $\beta = \frac{1}{k_b T}$, k_B est la constante de Boltzmann et T est la température. Dans la limite $T \to 0$, les seuls termes non-nuls de la somme sont $e^{-\beta E_0}$, c'est-à-dire les termes associés aux états fondamentaux du système. Ainsi, pour un système dégénéré de N_0 états fondamentaux, la fonction de partition devient $Z = N_0$. Calculer la fonction de partition correspond ainsi à déterminer le nombre d'états fondamentaux.

Le développement du calcul quantique a aussi mené à de fructueux aboutissements. L'algorithme de comptage quantique prend avantage de l'algorithme de Grover et de l'algorithme d'estimation de phase quantique pour approximer le nombre de solutions à l'aide de $(\sqrt{\frac{N}{M}})$ itérations, où M est le nombre de solutions, de taille exponentielle pour les problèmes SAT et N est le nombre d'états possibles [39]. Le nombre d'itérations nécessaires profite ainsi d'un gain quadratique par rapport à la complexité optimale classique de $O(\frac{N}{M})$.

Notons que dans tous les cas, le comptage de solutions aux formules CNF, l'objet de ce travail, ne s'effectue jamais en temps polynomial. Même les approches approximatives nécessitent un nombre exponentiel d'opérations, indiquant davantage la difficulté de ce problème.

1.5 Transitions de phase

La complexité d'une instance aléatoire d'un problème NP n'est pas toujours équivalente. En réalité, certaines instances sont résolubles en temps polynomial, alors que d'autres sont intraitables. Les discussions autour des classes de complexité s'intéressent principalement à la complexité dans le pire des cas pour décrire la difficulté inhérente d'un problème. Cependant, certaines instances peuvent posséder une complexité inférieure, pouvant alors être pris en avantage par certains algorithmes.

Pour les problèmes SAT, la difficulté d'une instance aléatoire est grandement dépendante sur le rapport $\alpha = \frac{m}{n}$, où m est le nombre de clauses et n est le nombre de variables. Intuitivement, cette difficulté provient de l'ajout de contraintes au problème sous la forme de clauses. Plus surprenamment, la complexité d'une telle instance suit une transition de phase nommée *transition de phase critique*. En effet, il existe une valeur critique α_c où les instances du problème SAT passe de satisfaisable à insatisfaisable dans la limite asymptotique de n. La difficulté du problème SAT est maximale juste avant cette transition. Pourquoi est-ce le cas ?

Plusieurs autres transitions de phase existent avant la transition critique telle qu'illustré à la figure 1.2. Ces transitions, similairement aux transitions de phase

du modèle d'Ising, correspondent à des changements dans l'organisation de l'ensemble des solutions. Pour décrire cette réorganisation, une mesure de similarité est nécessaire pour comparer les différentes entrées au problème SAT. La *distance de Hamming* entre deux chaînes de bits de même taille correspond au nombre de positions dans les chaînes où les bits correspondants diffèrent.

Avant la première transition de phase, la *transition de phase de regroupement*, toutes les solutions d'une instance du problème SAT sont situées au sein du même amas, chacune à une distance d'Hamming polynomiale selon la taille de l'instance. *Compléter.*

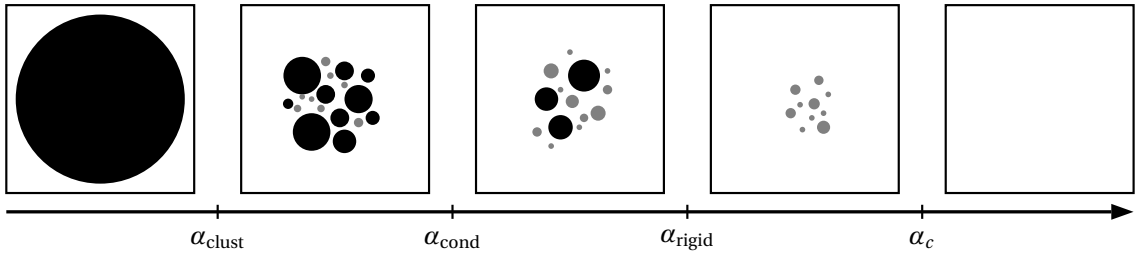

FIGURE 1.2 – Schéma des transitions de phase du problème SAT selon le rapport du nombre de clauses au nombre de variables α. Les transitions de phases illustrées sont la transition de regroupement α_{clust}, la transition de condensation α_{cond}, la transition de congélation α_{rigid} et la transition de satisfaisabilité α_c.

Les approches locales semblent échouer à la transition de condensation, alors que la difficulté du problème SAT semble provenir de la transition de congélation. La transition de phase critique pour NAE3SAT est situé à $\alpha_c \approx 2.1$ [40] et à $\alpha_c \approx 2/3$ [41] pour 1-in-3SAT. Les instances du problème #1-in-3SAT près du seuil critique appartiennent à la catégorie des problèmes bloqués. Les problèmes #P dans ce régime sont parmi les plus difficiles [42].

Chapitre 2

Échantillonnage quasi uniforme et comptage approximatif randomisé

Les problèmes de décision, ou équivalemment d'existence, et de comptage tel qu'introduits dans le chapitre précédent ne forment qu'une infime partie des problèmes étudiés dans la théorie de la complexité. Ces problèmes se cadrent notamment parmi les problèmes suivants :

(1) **Existence** : Existe-t-il une solution au problème ?
(2) **Construction** : Construire une solution au problème.
(3) **Génération uniforme** : Générer uniformément et aléatoirement une solution au problème.
(4) **Comptage** Combien de solutions satisfont le problème ?

Il n'est pas évident de prime abord de comparer la complexité de ces problèmes. Toutefois, dans une publication marquante pour le domaine de la complexité du comptage, Jerrum, Valiant et Vazirani offrent une unique perspective sur l'écart de complexité entre ces différents problèmes [3]. Les auteurs suggèrent d'abord que la génération uniforme est strictement plus difficile que la construction, et donc que l'existence comme la construction est évidemment au moins aussi difficile que l'existence. De plus, ils avancent que la génération uniforme est plus simple que le comptage, tout en offrant une réduction de la génération uniforme au comptage approximatif. En relaxant le comptage approximatif au comptage approximatif

randomisé, ils établissent une correspondance entre ce dernier et la génération quasi uniforme.

Ce travail éclaircit la compréhension de la théorie de la complexité, particulièrement pour la génération uniforme et du comptage. La dernière correspondance entre le comptage approximatif randomisé et la génération quasi uniforme s'avère particulièrement intéressante et forme la base du travail de ce mémoire. Cette inter-réduction mène en effet à un algorithme de comptage approximatif résolvant approximativement certains problèmes de comptage, possédant la propriété d'auto-réductibilité, à l'aide d'un générateur de solutions quasi uniforme. Par souci de commodité, cet algorithme est surnommé *algorithme de JVV* selon le nom de ses auteurs. Parmi ses principaux attraits est sa capacité à obtenir un compte approximatif avec un nombre polynomial d'appels au générateur de solutions, sous la condition que ce dernier soit génère une distribution de solutions suffisamment uniforme. Cette propriété s'avère particulièrement utile pour les problèmes possédant un nombre exponentiel de solutions comme le problème SAT.

Les trois concepts nécessaires à l'algorithme de JVV sont formalisés dans cette section : l'auto-réductibilité à la section 2.1, l'échantillonnage quasi-uniforme à la section 2.2 et le comptage approximatif randomisé à la section 2.3. Pour conclure, la section 2.4 présente une description en profondeur de l'algorithme de JVV.

2.1 Auto-réductibilité

Avant d'introduire le concept d'auto-réductibilité, il est utile de définir une notion supplémentaire. Les problèmes mentionnés dans l'introduction du chapitre sont souvent définis sur des objets mathématiques discrets, comme les formules propositionnelles, les graphes, les permutations, et bien plus. Ces objets se regroupent sous une même ombrelle, nommée *structure combinatoire*, définie comme un système discret fini comportant des éléments et des relations bien définies. Par exemple, une formule propositionnelle, composée de variables booléennes et de contraintes sous forme d'opérateurs booléens, est associée à un ensemble discret de solutions. Conséquemment, le domaine de l'*énumération combinatoire* cherche à déterminer le nombre d'éléments satisfaisant la relation. Pour le problème SAT, cela consiste

à trouver le nombre d'assignements satisfaisant la formule propositionnelle. De manière similaire, l'*optimisation combinatoire* tente de trouver la meilleure solution possible au problème.

L'*auto-réductibilité* (« self-reducibility ») est un concept complexe, essentiel à la compréhension du calcul et de la complexité, découlant de l'introduction de la *réduction automatique* (« autoreducibility ») [43, 44]. Ces concepts sont particulièrement importants dans le contexte de génération aléatoire et du comptage approximatif, mais aussi pour la réduction entre le problème de décision et les problèmes de recherche.

Définition 2.1 – Réduction automatique

Un problème algorithmique est dit *automatiquement réductible* s'il peut être résolu par un algorithme résolvant d'autres instances du même problème, sans que l'algorithme puisse interroger l'instance particulière qu'il cherche à résoudre.

Les problèmes automatiquement réductibles contiennent de l'information d'appartenance redondante, c'est-à-dire qu'il existe une structure dans l'ensemble de problèmes pouvant être exploitée pour simplifier le calcul d'une instance donnée. Ainsi, un algorithme peut résoudre une instance en utilisant l'information redondante présente dans d'autres instances, évitant ainsi les requêtes directes à l'instance en question. Connaître la solution à une autre problème peut alors simplifier la résolution du problème initial.

Pour discuter de la génération aléatoire et le comptage approximatif, l'*auto-réductibilité descendante*, une forme limitée de la réduction automatique, est une définition plus adéquate. En effet, cette condition est nécessaire à l'application de l'algorithme JVV.

Définition 2.2 – Auto-réductibilité descendante

Un problème algorithmique est dit *auto-réductible descendant* s'il peut être résolu grâce à un algorithme résolvant des instances de taille strictement inférieure.

Cette propriété s'éclaircit en prenant le problème SAT comme exemple. L'auto-réductibilité appliquée au problème de satisfaisabilité s'exprime facilement avec la

relation suivante [45] :

Relation 2.1 – Auto-réductibilité pour les problèmes SAT

Soit une constante $n \geq 1$ et une formule propositionnelle $\varphi(x_1, x_2, \ldots, x_n)$ où $x_i \in \{0, 1\}$. Alors,

$$\varphi(x_1, x_2, \ldots, x_n) = 1 \iff \varphi(x_1 = 0, x_2, \ldots, x_n) = 1 \vee \varphi(x_1 = 1, x_2, \ldots, x_n) = 1$$

Cette relation implique que l'ensemble de solutions d'une instance donnée peut être exprimée comme l'ensemble de solutions de deux instances plus petites du problème. Supposons que l'on souhaite résoudre une instance du problème SAT décrit par la formule CNF $\varphi(x_1, x_2, \ldots, x_n)$. Soit $\varphi_0 = \varphi(x_1 = 0, \ldots, x_n)$ et $\varphi_1 = \varphi(x_1 = 1, \ldots, x_n)$ deux sous-instances de l'instance φ, où la variable x_1 est remplacée par 0 et 1 respectivement. Ce faisant, la formule φ est alors raccourcie, comme certaines clauses sont satisfaites ou du moins réduites. Pour que l'instance φ soit satisfaisable, il est nécessaire qu'au moins une des sous-instances φ_0 et φ_1 soit satisfaisable. Dans le cas contraire, l'instance originale φ ne peut être satisfaisable car il n'existe pas de solutions peu importe la valeur de x_1. Ainsi, il suffit de considérer la disjonction des sous-problèmes possibles du problème original pour résoudre ce dernier. Les sous-problèmes obtenus étant aussi des formules propositionnelles, ceux-ci peuvent aussi être décomposés en problème de taille inférieure récursivement. Notons qu'il n'est pas nécessaire de construire la relation 2.1 avec la variable x_1, n'importe quelle variable x_i peut aussi être utilisée.

La relation 2.1 exemplifie une deuxième façon de voir l'auto-réductibilité : la résolution partielle d'une instance d'un problème laisse une plus petite instance du même type de problème. Une interprétation alternative est qu'une instance peut être résolue en résolvant des plus petites instances et en assemblant les sous-instances ensemble, tel l'assemblage d'un casse-tête.

Afin d'obtenir une meilleure perspective sur la notion d'auto-réductibilité, il est pertinent d'exprimer la structure d'une relation auto-réductible φ sous la forme d'un arbre orienté, nommé *arbre d'auto-réductibilité*, illustré à la figure 2.1. Dans cet arbre, les sommets représentent à la fois une chaîne de bits w de taille m et une instance de problème φ_w, où φ_w représente la sous-instance du problème φ dont

les premières variables sont remplacées par la chaîne de bits w. Le sous-problème est alors dit *fixé* par le préfixe w. Les arrêtes du graphe représentent l'assignement d'une variable. La racine de l'arbre correspond à l'instance du problème initial φ ainsi qu'à une solution partielle nulle. Les enfants de cette racine sont par la suite donnés par les sous-instances φ_w pour tous les préfixes w de taille 1 possibles. Le reste de l'arbre est défini récursivement de la même manière en augmentant la taille de la chaîne de bits w à chaque niveau jusqu'aux feuilles de l'arbre. Ces feuilles représentent finalement une entrée au problème φ, étant soit une solution ou une non-solution au problème.

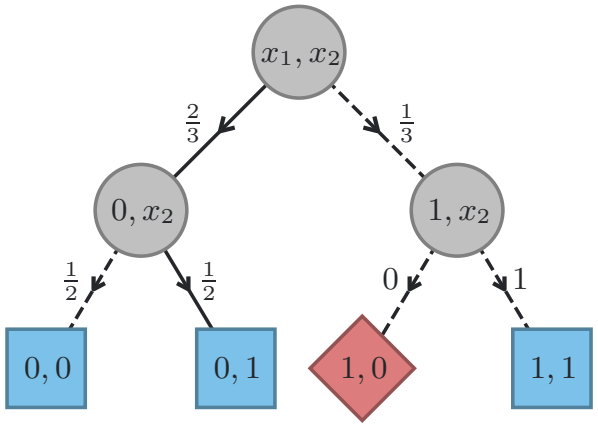

FIGURE 2.1 – *Refaire la figure.*

L'auto-réductibilité est simplement généralisable aux problèmes dont les variables peuvent prendre d valeurs, dit auto-réductibilité d-disjonctive, en prenant la disjonction sur les d valeurs possibles. Le degré de l'arbre est en conséquence donné par d. Notons aussi que l'arbre d'auto-réductibilité est aussi parfois défini tel les feuilles correspondent uniquement à des solutions au problème auto-réductible. Dans ce cas, les non-solutions ne sont pas incluses dans les ramifications de l'arbre.

Par complétude, une définition rigoureuse de l'auto-réductibilité dans le sens de Schnorr [46] est donnée. Celle-ci permet l'introduction d'un langage pertinent pour le reste de ce chapitre.

Définition 2.3 – Auto-réductibilité descendante

Soit Σ^* un ensemble fixe et fini encodant les instances d'un problème ainsi que leurs solutions. Soit $R \subseteq \Sigma^* \times \Sigma^*$ une relation binaire assignant à chaque instance de problème $x \in \Sigma^*$ un ensemble de solutions $\{ y \in \Sigma^* : xRy \}$. Une relation $R \subseteq \Sigma^* \times \Sigma^*$ est auto-réductible si et seulement si

(1) il existe une fonction calculable en temps polynomial $g \in \Sigma^* \to \mathbb{N}$ tel que $xRy \implies |y| = g(x)$;

(2) il existe une fonction calculable en temps polynomial $\psi \in \Sigma^* \times \Sigma^* \to \Sigma^*$ et $\sigma \in \Sigma^* \to \mathbb{N}$ satisfaisant

$$\begin{aligned} &\sigma(x) = O(\log |x|) \\ &g(x) > 0 \Rightarrow \sigma(x) > 0 \quad \forall x \in \Sigma^\star \\ &|\psi(x,w)| \leqslant |x| \quad \forall x, w \in \Sigma^\star \end{aligned}$$

et tel que, pour tout $x \in \Sigma^*, y = y_1 \dots y_n \in \Sigma^*$,

$$\langle x, y_1 \dots y_n \rangle \in R \Leftrightarrow \left\langle \psi \left(x, y_1 \dots, y_{\sigma(x)} \right), y_{\sigma(x)+1} \dots y_n \right\rangle \in R$$

Démystifions cette définition. Le problème SAT est un exemple de relation binaire R, où x encode une formule booléenne et y encode un assignement satisfaisable parmi l'ensemble des entrées possibles Σ^* tel que

$$\begin{aligned} R = \{ \langle x, y \rangle : x \in \Sigma^* \text{ encode une formule booléenne } B, \\ y \in \Sigma^* \text{ est un assignement satifaisable de } B \} . \end{aligned} \tag{2.1}$$

La notation xRy signifie que y est une solution valide à l'instance x. La première condition implique que la taille des solutions, donnée par la fonction g, est bornée par une fonction calculable en temps polynomial. La deuxième condition implique que, pour une instance x et un préfixe w de taille $g(x)$ de n'importe quelle solution y au problème x, ψ donne une instance x' dont les solutions sont exactement celles qui, lorsque concaténées avec w, forme les solutions de x. La fonction σ donne alors la granularité des solutions, c'est-à-dire le nombre de caractères de y utilisés pour réduire l'instance x. Plus précisément, $\psi(x,w) \leq |x|$ assure que la taille du problème

diminue à mesure que la réduction se poursuit. L'implication $g(x) > 0 \implies \sigma(x) > 0$ signifie que la réduction continue si la taille de l'instance x est non-nulle. La dernière relation énonce que la résolution du problème pour $\langle x, y \rangle$ peut être réduite à la résolution d'instances de taille inférieure, c'est-à-dire que $\langle x, y_1 \dots y_n \rangle$ est reliée à $\langle \psi(x, y_1 \dots y_n), y_{\sigma(x)+1} \dots y_n \rangle$.

Tous les problèmes NP-complet sont auto-réductibles [47], mais ce n'est pas le cas pour tous les problèmes de la classe NP [48]. De nombreux problèmes NP sont aussi auto-réductible descendant. Ces observations impliquent que l'algorithme de JVV s'applique en théorie à de nombreux problèmes intéressants. Toutefois, trouver une relation auto-réductible en pratique n'est pas toujours évident.

La définition de l'auto-réductibilité utilisée dans cette section s'applique principalement à l'équivalence entre l'échantillonnage quasi uniforme et le comptage approximatif randomisé. Cependant, une définition alternative [47] est souvent introduite pour l'équivalence entre les problèmes de décision et les problèmes de recherche, où un problème de recherche ne demande pas de montrer l'existence d'une solution, mais de construire une telle solution. En effet, si un problème est auto-réductible à ce sens, alors le problème de recherche est de la même complexité que le problème de décision.

2.2 Échantillonnage quasi uniforme

L'*échantillonnage uniforme* de structures combinatoires consiste à générer aléatoirement des solutions de manière uniforme, c'est-à-dire avec une probabilité suivant la distribution de probabilité uniforme des solutions, à un problème donné. Le problème de génération uniforme est assez méconnu, mais possède plusieurs applications dont la construction d'éléments représentatifs d'un ensemble, la formulation de conjectures pour un ensemble ou la vérification fonctionnelle du matériel informatique [49, 50].

La génération uniforme étant une demande plutôt rigide, il est parfois nécessaire de relaxer la condition d'uniformité pour une quasi-uniformité. Une distribution quasi uniforme est en pratique indiscernable d'une distribution uniforme, mais ce

relâchement facilite certaines preuves. La *distance en variation totale* est une mesure de distance statistique définie entre deux distributions de probabilité, déterminant la similitude entre ces distributions. Cette mesure permet entre autres de caractériser à quel point une distribution quasi uniforme se rapproche de la distribution uniforme.

Définition 2.4 – Distance en variation totale

Soit deux distributions de probabilité P et Q définies sur un ensemble dénombrable χ. La distance en variation totale est

$$\|P - Q\|_{TV} \equiv \frac{1}{2} \sum_{x \in \chi} |P(x) - Q(x)| .$$

Une distribution est alors dite quasi uniforme si sa distance en variation totale avec la distribution uniforme est suffisamment petite. Une définition commune demande que cette distance diminue proportionnellement avec l'inverse de la taille du problème. Un *échantillonneur quasi uniforme de tolérance* ξ génère alors des solutions selon une distribution de probabilités où la distance en variation totale entre celle-ci et la distribution uniforme est plus petite que ξ [51].

Définition 2.5 – Échantillonneur quasi uniforme

Un échantillonneur quasi uniforme pour un ensemble de solutions $S \subseteq \Sigma^* \times \Sigma^*$, où S représente la relation entre les instances d'un problème x et de ses solutions $y \in S(x)$, est un algorithme aléatoire prenant en entrée une instance $x \in \Sigma^*$ et une tolérance d'échantillonnage $\xi > 0$ et qui génère une solution $y \in S(x)$ tel que

$$\|Y - U\|_{TV} \leq \xi$$

où Y est la distribution de probabilité de y et U est la distribution de probabilité uniforme sur $S(x)$. Si l'algorithme s'exécute en temps borné par un polynomial en $|x|$ et en $\ln(\xi^{-1})$, on parle d'échantillonneur quasi uniforme pleinement polynomial (« Fully Polynomial Almost Uniform Sampler») (FPAUS).

Une notion supplémentaire est introduite dans ce mémoire afin de simplifier la notation. La *non-uniformité*, de manière similaire à la distance en variation totale, décrit la distance entre une distribution de probabilité Q et la distribution de

probabilité uniforme U.

Définition 2.6 – Non-uniformité

Soit la distribution de probabilité P et la distribution de probabilité uniforme U tel que $U(x) = 1/|x|$ définies sur un ensemble dénombrable χ. La non-uniformité est

$$\eta \equiv \frac{1}{2} \sum_{x \in \chi} |P(x) - U(x)| .$$

Un échantillonneur quasi uniforme de tolérance ξ possède en conséquence une non-uniformité plus petite que ξ.

2.3 Comptage approximatif randomisé

Comme mentionné au chapitre 1, les problèmes de comptage sont d'une grande complexité et nécessitent en pratique des méthodes approximatives pour la résolution de problèmes de grandes tailles. Le *comptage approximatif randomisé* simplifie davantage les attentes en ne demandant une solution approximative valide qu'avec une certaine probabilité. Un *schéma d'approximation randomisé de tolérance* ε se définit de manière similaire à l'algorithme d'approximation pour l'optimisation décrit à la section 1.3.

Définition 2.7 – Schéma d'approximation randomisé

Un schéma d'approximation randomisé pour un problème de comptage $f : \Sigma^* \to \mathbb{N}$ est un algorithme randomisé prenant en entrée une instance d'un problème $x \in \Sigma^*$ et une tolérance d'erreur $\varepsilon > 0$ et qui génère un nombre $N \in \mathbb{N}$ tel que, pour toute instance x,

$$\Pr\left[(1+\varepsilon)^{-1} f(x) \leq N \leq (1+\varepsilon) f(x)\right] \geq \frac{3}{4}.$$

Si l'algorithme s'exécute en temps borné par un polynomial en $|x|$ et ε^{-1}, alors on parle de schéma d'approximation randomisé pleinement polynomial (« Fully Polynomial Randomized Approximation Scheme ») (FPRAS).

La valeur $3/4$ est un nombre arbitraire pouvant être remplacée par n'importe quel nombre dans l'intervalle ouvert $(\frac{1}{2}, 1)$. Ce nombre, représentant la confiance de succès du schéma, peut être augmenté à n'importe quelle valeur $1 - \delta$ pour $\delta > 0$ au coût d'un facteur multiplicatif $O(\log(\delta^{-1}))$. Pour ce faire, il suffit de répéter le schéma original $O(\log(\delta^{-1}))$ fois et de prendre la médiane des résultats obtenus.

2.4 Algorithme de Jerrum-Valiant-Vazirani

Le travail de Jerrum, Valiant et Vazirani [3] établit une correspondance entre l'échantillonnage quasi uniforme et le comptage approximatif randomisé. Pour ce faire, deux algorithmes permettant le passage d'un problème à l'autre sont présentés. Bien la réduction du comptage à l'échantillonnage est intéressante, elle ne sera discuté que brièvement à la fin de la section comme la réduction d'importance est ici son inverse. L'algorithme de comptage approximatif randomisé, surnommé *algorithme de JVV*, permet de trouver le nombre de solutions d'un problème auto-réductible de la classe #P à une erreur multiplicative près avec un nombre polynomial d'appels à un générateur de solution quasi uniforme. Le comptage approximatif randomisé est utilisé plutôt que le comptage approximatif, car en raison de l'impossibilité d'obtenir un résultat déterministe à partir d'une procédure aléatoire.

Théorème 2.1 – Algorithme de JVV

Si un problème algorithmique auto-réductible dans #P de taille n admet un générateur de solutions avec une non-uniformité $\eta = O(1/n)$, alors le nombre de solutions peut être approximé à une erreur multiplicative $O(\eta n)$ avec une haute probabilité en utilisant un nombre polynomial d'appels au générateur.

Plus précisément, en exigeant un échantillonneur quasi uniforme pleinement polynomial, l'algorithme de JVV fournit un schéma d'approximation randomisé pleinement polynomial pour le comptage approximatif de relations auto-réductibles. Plus précisément, $O(\frac{n^2 \log \delta^{-1}}{\varepsilon^2})$ échantillons sont nécessaires.

Comment est-ce que l'algorithme de JVV opère ? Commençons par introduire l'idée principale derrière l'algorithme pour ensuite donner un exemple simple et compléter avec une description plus formelle.

Pour mieux comprendre son exécution, l'arbre d'auto-réductibilité, présenté à la section 2.1, est employé. L'algorithme de JVV implique de se déplacer vers le haut ou vers le bas de cet arbre, en s'intéressant à toutes les sous-instances en chemin. Supposons qu'il est possible d'échantillonner uniformément les solutions de n'importe quelle instance du problème SAT (la généralisation à un échantillonneur quasi uniforme est effectuée plus bas). Commençons d'abord par échantillonner des solutions au problème initial φ représenté par la racine de l'arbre. Soit P_0 et P_1 les probabilités que la première variable x_1 soit respectivement de 0 et 1 pour les solutions échantillonnées. Ces probabilités reflètent la proportion de solutions dans chaque sous-arbre. Par la suite, descendons l'arbre vers l'enfant le plus probable φ_{w_1}, représentant la formule φ réduite par le préfixe w_1, en gardant en mémoire la probabilité P_{w_1}. Comme notre échantillonneur peut aussi résoudre le problème subséquent, nous pouvons encore déterminer la valeur la plus probable de la variable suivante x_2 en déterminant la probabilité P_{w_2}. Le processus est répété jusqu'à atteindre les feuilles de l'arbre, en sauvegardant les probabilités P_{w_i} le long du chemin w choisi. Au dernier noeud avant les feuilles, nous obtenons une solution avec une probabilité P_{w_n}. Comme nous sommes à la base de l'arbre, cette probabilité indique le nombre de solutions que possède la dernière sous-instance. En effet, ce sous-arbre contient nécessairement $\frac{1}{P_{w_n}}$ solutions, car il y a nécessairement une ou deux solutions à la dernière sous-instance. En remontant l'arbre, on remarque que la sous-instance précédente contient $\frac{1}{P_{w_{n-1}}}$ fois plus de solutions. Par conséquent, le nombre de solutions dans cette partie de l'arbre est de $\frac{1}{P_{w_n}}\frac{1}{P_{w_{n-1}}}$. Continuant cette procédure jusqu'à la racine, nous trouvons que le nombre de solutions est donné par

$$N = \frac{1}{P_{w_0}}\frac{1}{P_{w_m}}\cdots\frac{1}{P_{w_n}} = \frac{N}{N_{w_0}}\frac{N_{w_0}}{N_{w_1}}\cdots\frac{N_{w_{n-1}}}{N_{w_n}}. \tag{2.2}$$

Éclaircissons maintenant cette à l'aide d'un exemple simple illustré à la figure 2.2. Soit une instance du problème SAT décrite par la formule CNF $\varphi(x_1, x_2) = \neg x_1 \wedge x_2$, dont les solutions sont les couples $(0,0)$, $(0,1)$ et $(1,1)$. Un arbre est construit de ma-

nière à représenter toutes les sous-instances de l'instance originale. Chaque niveau de l'arbre correspond à l'attribution d'une valeur, 0 ou 1, à une variable, constituant ainsi une sous-instance du problème initial. La racine de l'arbre représente le problème initial et ses deux variables x_1 et x_2. La première couche représente les sous-problèmes φ_0 et φ_1 où la première variable est remplacée par 0 et 1 respectivement. Les feuilles représentent les assignations possibles au problème φ, c'est-à-dire lorsque toutes les variables sont fixées à une certaine valeur. Les arrêtes de l'arbre partant des noeuds du niveau i au niveau $i+1$ décrivent la probabilité qu'une solution au problème commence avec le préfixe w_i. Maintenant que l'arbre d'auto-réductibilité est construit, comptons le nombre de solutions à la formule $\varphi(x_1, x_2)$. En échantillonnant l'instance originale, nous trouvons que les échantillons possèdent une probabilité $P_0 = \frac{2}{3}$ de commencer avec la variable $x_1 = 0$. La sous-instance $\varphi_0 = \varphi(0, x_1)$ est donc choisi, comme il s'agit de l'enfant le plus probable, et de nouvelles solutions sont générées. Nous remarquons que celles-ci possèdent la même probabilité $P_w = \frac{1}{2}$ de commencer par le préfixe $w = 00$ que $w = 01$. Après avoir choisi le préfixe $w = 01$, le compte est obtenu de la façon suivante :

$$N = \frac{1}{P_0}\frac{1}{P_{01}} = 3 \tag{2.3}$$

Quelques notes sont ici nécessaires. Le chemin des probabilités maximales est choisi comme une meilleure précision est atteinte avec un plus grand nombre de solutions. De plus, ce choix évite d'obtenir une probabilité nulle, c'est-à-dire lorsque le sous-arbre ne possède aucune solution. Jusqu'à maintenant, le générateur a été considéré comme uniforme. Il est cependant possible d'utiliser une distribution quasi uniforme comme montré ci-bas.

Supposons que le nombre exact de solutions pour une instance de problème SAT de n est de N. Notons $x_{:m}$ le préfixe de taille m de la chaîne de bits x et $N_{x_{:m}}$ le nombre de solutions commençant avec le préfixe $x_{:m}$. Par la suite, supposons que nous avons accès à un générateur qui échantillonne l'ensemble de solutions uniformément et aléatoirement, tel que chaque solution est échantillonnée avec une probabilité $p^\star = \frac{1}{N}$. Pour n'importe quelle solution z, on écrit

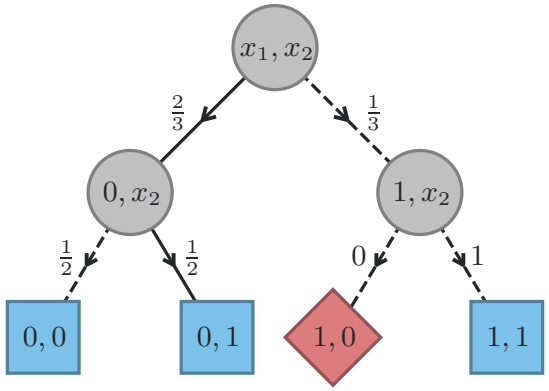

FIGURE 2.2 – Exemple de l'arbre d'auto-réductibilité dans l'algorithme de JVV pour une formule CNF $\varphi(x_1, x_2)$, où les solutions sont illustrées par un carré bleu et les non-solutions par un losange rouge. Suivant le chemin des probabilités maximales, le compte approximatif est donné par $N = \frac{1}{P_0}\frac{1}{P_{01}} = 3$.

$$p^\star = \frac{1}{N} = \frac{N_{z_{:1}}}{N} \cdot \frac{N_{z_{:2}}}{N_{z_{:1}}} \cdots \frac{N_{z_{:n}}}{N_{z_{:n-1}}} \tag{2.4}$$

$$= p(z_{:1}) \cdot p(z_{:2}|z_{:1}) \cdots p(z_{:n}|z_{:n-1}) \tag{2.5}$$

$$= p(z_{:1}) \prod_{i=1}^{n-1} p(z_{:i+1}|z_{:i})\,, \tag{2.6}$$

où $N_{z_{:n}} \equiv 1$. La probabilité conditionnelle $p(z_{:i+1}|z_{:i}) \equiv N_{z_{:i+1}}/N_{z_{:i}}$ est la probabilité qu'une solution échantillonnée z' soit $z'_{i+1} = z_{i+1}$ sachant que $z'_{:i} = z_{:i}$.

L'algorithme de JVV pour le comptage approximatif retourne une approximation à un facteur multiplicatif près de $p^\star$, et donc de N, en approximant les probabilités conditionnelles $p(z_{:i+1}|z_{:i})$. Cela est fait en générant un ensemble d'échantillons S et en approximant les probabilités conditionnelles avec

$$\tilde{p}(z_{:i+1}|z_{:i}) = \frac{\tilde{N}_{z_{:i+1}}}{\tilde{N}_{z_{:i}}}\,, \tag{2.7}$$

où $\tilde{N}_{z_{:i}}$ est le nombre de solutions dans l'ensemble d'échantillons commençant

par $z_{:i}$. Donc,

$$\tilde{p}^{\star} = \tilde{p}(z_{:1}) \prod_{i=1}^{n-1} \tilde{p}(z_{:i+1}|z_{:i}) \approx \frac{1}{N}\,. \tag{2.8}$$

Le nombre approximatif de solutions est ainsi obtenu. L'algorithme se résume par l'algorithme 2.4.

Algorithm 1: Algorithme de JVV

Require: Formule CNF : $f(n, m)$, Nombre d'échantillons : n_s

$w \leftarrow$ ""
$\tilde{N} \leftarrow 1$
for $i \in \{1, \ldots, n\}$ **do**
  $S \leftarrow \{\ \}$
  while $|S| < n_s$ **do**
    $m \leftarrow$ Échantillonnage$(f(n, m))$
    $S \leftarrow S \cup \{\ m\ \}$
  end while
  $w, \tilde{p} \leftarrow \arg\max_{w' \in \{\ w+0, w+1\ \}} |\{\ s \in S : s_{:|w'|} = w'\ \}|/|S|$
  $\tilde{N} \leftarrow \tilde{N}/\tilde{p}$
end for
return $\tilde{N}$

Chapitre 3

Algorithmes variationnels quantiques

En général, le comptage est un problème ardu. En acceptant une solution approximative, la complexité de ce problème peut être déplacée à l'échantillonnage quasi uniforme de solutions à ce problème grâce à l'algorithme de JVV. Toutefois, la construction d'un tel générateur n'a pas encore été évoquée. En fait, la majorité des solutionneurs de problème #P ne se basent pas sur l'échantillonnage en raison de la difficulté d'obtenir une distribution uniforme composée de solutions. Est-ce que le calcul quantique peut offrir une méthode efficace pour la génération uniforme de solutions ?

Les *algorithmes variationnels quantiques* (« Quantum Variational Algorithms ») (VQA) sont des algorithmes hybrides, c'est-à-dire composés d'une partie quantique et d'une partie classique, conçus pour exploiter les avantages du calcul quantique tout en profitant de la puissance des algorithmes classiques [2]. Ces algorithmes ont émergés comme la stratégie dominante pour atteindre l'avantage quantique avec le matériel informatique quantique actuel, connu sous le nom des ordinateurs quantiques bruités de taille intermédiaire (« Noisy Intermediate-Scale Quantum ») (NISQ). En effet, les algorithmes quantiques possédant un avantage par rapport aux algorithmes classiques sont présentement hors d'atteinte pour les ordinateurs quantiques du moment en raison de la taille des systèmes nécessaires et des erreurs causées par le bruit. Le concept derrière des VQA s'inspire des méthodes d'apprentissage automatique pour résoudre des problèmes d'optimisation combinatoire avec des circuits de faible profondeur sans se soucier de la correction des erreurs des

qubits. Un état quantique initial facile à préparer est évolué unitairement avec un circuit quantique paramétré et la valeur moyenne d'une fonction de coût est estimé par de multiples mesures du circuit dans une base appropriée. Les paramètres du circuit sont alors ajustés itérativement avec un optimiseur classique pour minimiser la fonction de coût et ainsi préparer un état près d'une superposition des solutions du problème. Cette approche limite les inconvénients causés par le bruit en raison de l'utilisation de circuits paramétrés limitant la taille des circuits utilisés. Ainsi, les VQA sont une option possible pour la résolution de problèmes d'échantillonnage de l'algorithme de JVV.

Cet chapitre introduit d'abord l'algorithme adiabatique quantique à la section 3.1, fondant la base théorique de QAOA. Après avoir détaillé QAOA à la section 3.2, différentes variantes de QAOA sont explorées, tel comme sa généralisation, l'ansatz quantique opérateur alternées, et une variante de celle-ci utilisant le forçage de Grover à la section 3.3. Finalement, deux propriétés de QAOA sont explorés, c'est-à-dire l'initialisation et l'optimisation des paramètres du circuit quantique à la section 3.4 et le biais d'échantillonnage à la section 3.5.

3.1 Algorithme adiabatique quantique

Le *théorème adiabatique*, introduit par Born et Fock [52], peut être énoncé simplement comme suit :

Théorème 3.1 – Théorème adiabatique

Un système physique demeure dans son état propre instantané si une perturbation donnée agit sur lui suffisamment lentement et s'il y a un intervalle significatif entre la valeur propre et le reste du spectre de l'hamiltonien.

Bien que différentes versions de ce théorème furent rigoureusement formulées [53], une version approximative de celui-ci, proposée par Messiah [54] et rectifiée par Amin [55], est présentée ici dans l'objectif d'élucider les mécanismes du théorème. Un système quantique, décrit par un hamiltonien dépendant du temps $H(t)$, évolue selon l'équation de Schrödinger

$$i\hbar\frac{\partial\,|\psi(t)\rangle}{\partial t} = H(t)\,|\psi\rangle\ .$$

Considérons ici que l'hamiltonien $H(t)$ peut s'écrire sous la forme $H(t) = \tilde{H}(s)$, où $s = t/T \in [0,1]$ est le temps adimensionnel, de manière que T contrôle le taux de variation dans le temps de $H(t)$. Soit $|\varepsilon_j(s)\rangle$ les états propres instantanés de $\tilde{H}(s)$ avec énergie ε_j (potentiellement dégénérée) tel que

$$\tilde{H}(s)\,|\varepsilon_j(s)\rangle = \varepsilon_j(s)\,|\varepsilon_j(s)\rangle\ , \tag{3.1}$$

où $\varepsilon_j(s) < \varepsilon_{j+1}(s)\ \forall j, s$ et $j \in \{\,0, 1, 2, \dots\,\}$. L'approximation adiabatique indique qu'un état initial préparé dans un des états propres instantanés $|\varepsilon_j(0)\rangle$ demeure dans le même état propre instantané $|\varepsilon_j(t)\rangle$ à une phase globale près si $\varepsilon_i(s) - \varepsilon_j(s) \neq 0$ et

$$T \gg \max_{s\in[0,1]} \frac{|\langle\varepsilon_i(s)|\partial_s\tilde{H}(s)|\varepsilon_j(s)\rangle|}{|\varepsilon_i(s) - \varepsilon_j(s)|^2}\ \forall j \neq i\,. \tag{3.2}$$

L'approximation adiabatique est souvent utilisée à partir de l'état fondamental $|\varepsilon_0(t)\rangle$, menant à la définition du gap spectral entre l'état fondamental et le premier état excité du système $\Delta(s) = \varepsilon_1(s) - \varepsilon_0(s)$. Généralement, le maximum de $\langle\varepsilon_i(s)|\partial_s\tilde{H}(s)|\varepsilon_j(s)\rangle$ est de l'ordre d'une valeur propre typique de $\tilde{H}$ et petit. Le minimum du carré de l'inverse du gap spectral Δ constitue alors un critère pratique pour quantifier le temps nécessaire à l'évolution adiabatique.

L'*algorithme adiabatique quantique* (« Quantum Adiabatic Algorithm ») (QAA), introduit par Farhi, Gutmann et Sipser [56], emploie un ordinateur quantique physique pour la résolution de problèmes d'optimisation combinatoire en se basant sur le théorème adiabatique quantique. Pour ce faire, le système physique est initialement préparé dans l'état fondamental d'un hamiltonien de forçage H_D facile à construire et dont l'état fondamental est simple à trouver. La solution du problème, encodée dans l'état fondamental de l'hamiltonien de problème H_P, est alors obtenue en transitionnant de l'état fondamental de l'hamiltonien H_D à l'état fondamental de

l'hamiltonien H_P par une évolution adiabatique. Plus précisément, l'hamiltonien du système s'écrit comme

$$\tilde{H}(s) = (1-s)\, H_D + s H_P \,. \tag{3.3}$$

Ainsi, en présumant que le gap spectral entre l'état fondamental et l'état excité est non-nul, la solution du problème est toujours obtenue à partir de l'état fondamental obtenu si l'évolution, donnée par l'opérateur $U(t) = e^{-i\int_0^1 \tilde{H}(s)ds}$, est suffisamment lente tel que garantit par le théorème adiabatique quantique. Si le temps d'évolution est trop petit, une *transition diabatique*, c'est-à-dire une transition de l'état fondamental à un état excité, peuvent empêcher l'évolution adiabatique.

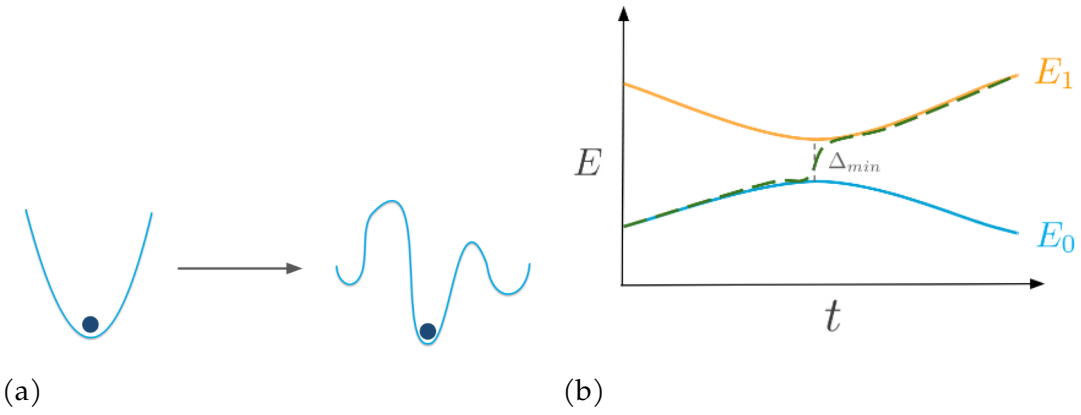

FIGURE 3.1 – *Refaire les figures.*

Typiquement, le gap spectral est non-nul [56], mais cela n'est pas suffisant pour rendre l'algorithme utile comme ce gap doit être assez grand pour limiter le temps d'évolution. Le gap spectral peut être suffisamment grand pour permettre une évolution adiabatique dans un temps réaliste pour certains problèmes, mais ce n'est pas toujours possible [57]. Une alternative consiste à trouver un compromis entre le temps d'évolution et la proximité de l'état final avec l'état fondamental espéré. Le choix du chemin adiabatique utilisé pour transitionner de H_D à H_P peut aussi différer de l'équation 3.3 dans l'objectif de maximiser le gap au long du chemin et donc minimiser le temps d'évolution [58, 59].

L'algorithme adiabatique quantique se place au sein du *calcul adiabatique quantique*, qui regroupe différentes méthodes similaires. Un autre membre de ce groupe est le *recuit quantique* qui représente généralement l'emploi de QAA dans un environnement bruité, menant ainsi à une version plus réaliste sans contrainte d'adiabaticité ou d'universalité. Cet algorithme partage plusieurs similitudes avec l'algorithme quantique d'optimisation approximative qui seront explorées dans les prochaines sections.

3.2 Algorithme quantique d'optimisation approximative

Bien que le calcul adiabatique quantique soit utilisé pour résoudre les problèmes d'optimisation combinatoire, le temps nécessaire pour une évolution adiabatique constitue un facteur limitant pour de nombreux problèmes. L'*algorithme quantique d'optimisation approximative* (« Quantum Approximate Optimization Algorithm ») (QAOA) [5] propose alors une alternative, sous la forme d'un algorithme variationnel quantique, discrétisant l'évolution continue de QAA. Bien que cette approche s'éloigne de l'évolution adiabatique, celle-ci prend avantage des transitions diabatiques entre l'état fondamental et les états excités pour réduire le temps d'évolution. L'optimisation des paramètres du circuit quantique paramétré permet de naviguer efficacement l'espace de Hilbert pour atteindre une bonne solution approximative. En fait, QAOA est contre-adiabatique et non seulement adiabatique, signifiant que des raccourcis à l'adiabacité sont possibles [60]

3.2.1 Description de l'algorithme

Étant une idée prometteuse pour les applications des ordinateurs quantiques, l'algorithme quantique d'optimisation approximative a mené à une quantité incroyable de travaux dans les précédentes années. Ainsi, pour simplifier la compréhension de ce concept, l'algorithme original, dû à Farhi, Goldstone et Gutmann [5], est d'abord présenté.

QAOA repose sur deux différents hamiltoniens : l'hamiltonien de problème, ou de phase, H_P et l'hamiltonien de forçage, ou de mélange, H_D. L'hamiltonien de problème est formulé de façon à encoder la solution, potentiellement dégénérée, du problème d'optimisation combinatoire dans son état fondamental. Pour ce faire, celui-ci est défini en fonction de la fonction de coût C de l'instance du problème : $H_P |x\rangle = C(x) |x\rangle$. H_P prend typiquement la forme de l'hamiltonien du modèle d'Ising. L'hamiltonien de forçage, quant à lui, est donné par

$$H_D^X = \sum_{i=1}^{n} X_i \,, \tag{3.4}$$

où X_i est l'opérateur de Pauli X appliqué sur le qubit i d'un système à n qubits. H_D^X est construit de manière à induire de l'interférence et ainsi permettre l'exploration de l'espace de Hilbert.

Par définition, l'hamiltonien H_p est diagonal dans la base computationnelle, alors que l'hamiltonien H_D comprend des termes hors diagonaux de sorte que ceux-ci ne commutent pas entre eux. Deux opérations unitaires paramétrées sont définies à partir des hamiltoniens H_P et H_D : l'opérateur de problème $U_P(\gamma) = e^{-i\gamma H_P}$ ainsi que l'opérateur de forçage $U_D(\beta) = e^{-i\beta H_D}$, où γ et β sont des paramètres réels. L'opérateur U_P représente une rotation de phase, paramétrisée par γ, des états de la base computationnelle en fonction de leur énergie donné par H_P. L'opérateur U_D, paramétrisé par β, superpose différents états de la base computationnelle ayant précédemment acquis différents facteurs de phase, menant ainsi à de l'interférence.

En tant que VQA, QAOA est un algorithme hybride composé d'un circuit quantique paramétré et d'un optimiseur classique. Le circuit quantique est d'abord préparé dans un état propre de l'hamiltonien de forçage. Pour l'hamiltonien 3.4, un exemple d'état initial possible est dans une superposition égale des états possibles $= |+\rangle^{\otimes n}$. Le produit des opérateurs $U_D U_P$ est alors appliqué en alternance p fois sur l'état initial $|\psi_0\rangle$, donnant l'état suivant :

$$|\psi(\vec{\gamma}, \vec{\beta})\rangle = \underbrace{U_D(\beta_p)U_P(\gamma_p)\cdots U_D(\beta_1)U_P(\gamma_1)}_{p \text{ fois}} |\psi_0\rangle \,, \tag{3.5}$$

où $\vec{\gamma} = (\gamma_1, \ldots, \gamma_p)$ et $\vec{\beta} = (\beta_1, \ldots, \beta_p)$ sont les paramètres initiaux du circuit. Une fois l'état $|\psi(\vec{\gamma}, \vec{\beta})\rangle$ préparé, la valeur moyenne de l'hamiltonien de problème H_P est calculé par des mesures répétées de l'état final dans la base computationnelle :

$$E_P(\vec{\gamma}, \vec{\beta}) = \langle \psi(\vec{\gamma}, \vec{\beta}) | H_P | \psi(\vec{\gamma}, \vec{\beta}) \rangle \ . \tag{3.6}$$

L'énergie trouvée quantifie l'optimalité de l'état préparé. Par la suite, une méthode d'optimisation classique continue, comme la descente de gradient stochastique, est employée pour mettre à jour itérativement les paramètres γ et β du circuit paramétré de manière à minimiser la valeur moyenne $E_P(\vec{\gamma}, \vec{\beta})$:

$$(\vec{\gamma}^*, \vec{\beta}^*) = \arg\min_{\vec{\gamma}, \vec{\beta}} E_P(\vec{\gamma}, \vec{\beta}) \, . \tag{3.7}$$

Si l'optimisation aboutit de manière espérée, l'état final $|\psi(\vec{\gamma}^*, \vec{\beta}^*)\rangle$ correspond à une superposition des états fondamentaux de H_P et donc aux différentes solutions du problème étudié. Si ce n'est pas le cas, le ratio d'approximation α est utilisé pour décrire la qualité de la solution trouvée comme à la section 1.3 :

$$\alpha_{\vec{\gamma}^*, \vec{\beta}^*} = \frac{E_p(\vec{\gamma}^*, \vec{\beta}^*)}{\min_{(\vec{\gamma}, \vec{\beta})} E_p(\vec{\gamma}, \vec{\beta})} \tag{3.8}$$

Ce ratio augmente théoriquement avec le nombre de couches p utilisées comme QAOA équivaut à une évolution adiabatique dans la limite $p \to \infty$ [5]. Notons que pour l'utilisation de QAOA sur des graphes, la valeur de p doit augmenter avec la taille du graphe pour ne pas être limité par la localité de QAOA [61].

L'algorithme se résume par les étapes suivantes, illustrées dans la figure 3.2 :

(1) Définition de l'hamiltonien de problème H_P.
(2) Préparation de l'état initial $|\psi_0\rangle$.
(3) Construction du circuit quantique paramétré $|\psi(\gamma, \beta)\rangle$ en appliquant en alternance les opérateurs $U_P(\gamma)$ et $U_D(\beta)$ p fois.
(4) Calcul de l'énergie E_P à travers de mesures dans la base computationnelle.

(5) Optimisation des paramètres $\vec{\gamma}$ et $\vec{\beta}$ à l'aide d'un optimiseur classique minimisant l'énergie E_P.

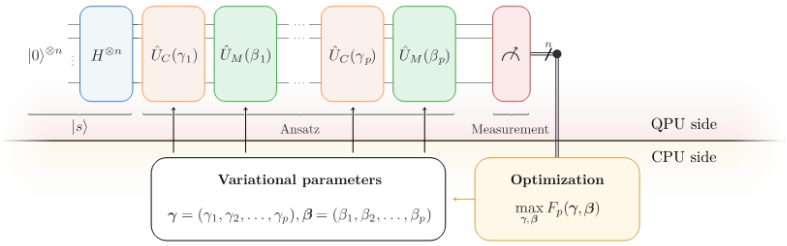

FIGURE 3.2 – *Refaire la figure.*

Décrivons maintenant plus en détails les mécanismes derrière QAOA : la préparation de l'état initial, l'encodage du problème dans un hamiltonien de problème ainsi que le choix de l'hamiltonien de forçage. L'initialisation et l'optimisation des paramètres du circuit est décrit à la section 3.4.

Préparation de l'état initial

Comme QAA, le circuit quantique est initialement préparé dans l'état fondamental de l'hamiltonien de forçage pour garantir le théorème adiabatique. Cependant, la divergence de QAOA du calcul adiabatique implique qu'un autre choix d'état initial peut être choisi. Une alternative consiste à utiliser une solution approximative du problème comme état initial [62].

Encodage du problème

Comment est-ce qu'un problème d'optimisation combinatoire $\varphi(x)$ peut être encodé par un hamiltonien H_P ? D'abord, les entrées x du problème sont caractérisées par une fonction de coût $C(x)$, composés termes de pénalisant les configurations non-valides. Une entrée optimale est associée à un coût nul, alors qu'une entrée non optimale est associée à un coût positif selon sa proximité avec les solutions. Résoudre le problème correspond alors à trouver l'entrée optimale, c'est-à-dire l'entrée x^* minimisant la fonction de coût. L'hamiltonien de problème se définit par

$$H_P |x\rangle = C(x) |x\rangle \tag{3.9}$$

Cette équation implique que l'état fondamental de l'hamiltonien H_P encode les solutions au problème donné. Un état préparé dans l'état fondamental de H_P correspond ainsi à une superposition des solutions au problème φ. Le modèle d'Ising est fréquemment utilisé pour fournir un hamiltonien décrivant la fonction de coût C :

$$H_P = -\sum_{(i,j)\in E} J_{ij}\sigma_i\sigma_j - \sum_{i\in V} h_i\sigma_i\,, \tag{3.10}$$

où E est l'ensemble d'arêtes, V est l'ensemble de sommets et σ_i est le spin de la particule i. Les constantes J et h représentent respectivement l'interaction entre deux sites et le champ magnétique externe. Le problème d'optimisation quadratique binaire non contraint (« Quadratic Unconstrained Binary Optimization ») (QUBO) permet aussi d'exprimer les fonctions de coût de manière plus générale, en ne les restreignant pas au modèle d'Ising.

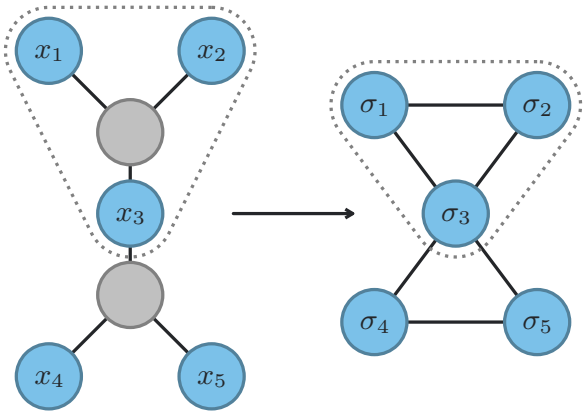

FIGURE 3.3 – Exemple de la transformation entre le modèle d'Ising pour NAE3SAT et 1-in-3SAT pour une formule φ. À gauche, le graphe de facteurs de φ avec les sommets de variables (bleus) et de clauses (gris). À droite, le modèle d'Ising correspondant à φ. Pour 1-in-3SAT, un terme de champ magnétique est ajouté pour favoriser les configurations où chaque clause contienne exactement un variable fixée à 1.

Comme trouver l'état fondamental d'un modèle d'Ising constitue un problème NP-complet, il existe alors nécessairement une réduction entre ce problème et les différents problèmes NP-complet. Bien que ces réductions ne sont pas nécessairement évidentes, plusieurs de celles-ci ont été formulées pour le calcul adiabatique

quantique [63, 64]. Une réduction intéressante dans notre cas relie le problème positif NAE3SAT au modèle d'Ising antiferromagnétique. Une formule CNF φ peut être représentée graphiquement sous la forme d'un graphe bipartie nommé *facteur de graphes*. Dans ce graphe, les clauses et les variables sont représentés sous la forme de sommet alors la présence des variables dans une clause sont représentées à l'aide d'arêtes entre les deux partitions. Considérons pour le moment une seule clause $C = (x_1 \vee x_2 \vee x_3)$ de φ. En associant chaque variable x_i à un spin σ_i selon la transformation $\sigma_i = 1 - 2x_i$, cette clause se réduit à un modèle d'Ising en considérant un réseau triangulaire composé des spins σ_i, où chacun de ceux-ci sont reliés aux deux autres spins tels qu'illustré à la figure 3.3. L'hamiltonien d'un tel modèle est donné par

$$H_{P,C} = \sigma_1\sigma_2 + \sigma_2\sigma_3 + \sigma_3\sigma_1 \tag{3.11}$$

Les énergies associées à chaque combinaison possible des spins sont présentés dans le tableau 3.1a, où les spins vers le haut sont représentés par 0 et les spins vers le bas sont représentés par 1. La frustration des spins sur le réseau implique que les seuls états qui ne sont pas dans l'état fondamental sont les états 000 et 111. Hors, il s'agit des deux combinaisons ne faisant pas partie de l'ensemble des solutions du problème positif NAE3SAT. L'état fondamental de l'hamiltonien $H_{P,C}$ correspond ainsi bien aux solutions de la clause C.

Entrée	Énergie
000	3
001	-1
010	-1
100	-1
011	-1
110	-1
101	-1
111	3

(a)

Entrée	Énergie
000	0
001	-2
010	-2
100	-2
011	0
110	0
101	0
111	6

(b)

Tableau 3.1 – Énergie de chaque entrée dans le modèle d'Ising pour le problème NAE3SAT (a) et 1-in-3SAT (b).

La formule est représentable par un modèle d'Ising en appliquant la transforma-

tion précédente sur chacune des clauses de la formule, tel qu'illustré à la figure 3.3, donnant ainsi l'hamiltonien

$$H_P = \sum_C H_{P,C}\,, \tag{3.12}$$

de manière que chaque clause soit respectée. Ainsi, l'hamiltonien de problème pour NAE3SAT est donné par l'équation 3.10 avec $J_{ij} = -1$ et $h_i = 0$. Le problème 1-in-3SAT se transforme au modèle d'Ising en ajoutant pour chaque clause l'hamiltonien

$$H_{P,C} = \sigma_1\sigma_2 + \sigma_2\sigma_3 + \sigma_3\sigma_1 - \sigma_1 - \sigma_2 - \sigma_3$$

Un champ magnétique externe est ajouté pour imposer la contrainte supplémentaire, c'est-à-dire que toutes les clauses doivent contenir exactement une variable évaluant à vrai. Le tableau 3.1b présente les énergies des configurations pour l'hamiltonien précédent. L'hamiltonien pour le problème 1-in-3SAT est donc donnée par l'équation 3.10 avec $J_{ij} = -1$ et $h_i = 1$.

Choix du forçage

L'hamiltonien de forçage H_D initialement proposé avec QAOA prends son inspiration de l'algorithme adiabatique quantique, en utilisant l'hamiltonien facile à préparer H_D^X. Cela implique que toute la dépendance au problème doit être encodée dans l'hamiltonien de problème H_P. Comme QAOA n'est pas restreint par cette condition, des opportunités se présentent pour encoder différemment le problème. Par exemple, l'Hamiltonien de forçage peut être utilisé pour restreindre l'espace de Hilbert en prenant en compte la structure du problème. Divers hamiltoniens offrent différentes performances selon le problème étudié. Le choix de forçage demeure encore une question ouverte.

3.2.2 Relation avec l'algorithme adiabatique quantique

QAA requiert une évolution continue de l'état, alors que QAOA repose sur l'application de portes quantiques. Pour établir un lien entre QAA et QAOA, l'évolution de l'hamiltonien doit donc être rendu discrète. Pour ce faire, la décomposition de Suzuki-Trotter, donnée par $e^{(A+B)t} = \lim_{n\to\infty}(e^{At/n}e^{Bt/n})^n$ pour deux opérateurs A et B, est employée. Le premier ordre de cette décomposition permet de discrétiser l'opérateur de l'hamiltonien dépendant du temps $H(t) = (1 - \frac{t}{T})H_D + \frac{t}{T}H_P$ de QAA (voir l'équation 3.3), où $t \in [0, T]$, tel que

$$e^{-i\Delta t H(t)} \approx e^{-i(1-\frac{t}{T})H_D\Delta t}e^{-i\frac{t}{T}H_P\Delta t} + O(\Delta t^2)\,, \tag{3.13}$$

où Δt est le pas de temps. Ainsi, l'évolution unitaire de QAA peut être émulée avec des portes quantiques en décomposant $U(t) = e^{-i\int_0^T H(t)dt}$ en séquence de petits pas de temps à l'aide de la formule de Suzuki-Trotter au premier ordre :

$$U(t) \approx \prod_{k=0}^{t/\Delta t-1} e^{-iH(k\Delta t)\Delta t} = \prod_{k=0}^{t/\Delta t-1} e^{-i(1-\frac{k\Delta t}{T})H_D\Delta t}e^{-i\frac{k\Delta t}{T}H_P\Delta t}\,.$$

Ce processus est généralement nommé *évolution adiabatique trottérisée* d'après la décomposition de Suzuki-Trotter. La forme de QAOA est alors retrouvée en remplaçant $(\frac{kt}{T})\Delta t$ et $(1 - \frac{kt}{T})\Delta t$ par les paramètres γ et β. Dans la limite $p \to \infty$, la condition adiabatique est satisfaite comme une évolution adiabatique trottérisé est retrouvée. Par contre, pour une profondeur de circuit p finie, plus les paramètres γ et β est petits, plus les erreurs dues à la décomposition, dites erreurs de Trotter, sont petites. Cependant, cela implique aussi que le temps d'évolution adiabatique T est plus petit et donc que des excitations diabatiques affectent négativement la performance de l'algorithme. Au contraire, si les paramètres sont grands, le temps d'évolution augmente aux dépens de l'impact des erreurs de Trotter. Un compromis entre ces deux facteurs doit être choisi.

3.3 Ansatz quantique à opérateurs alternants

L'algorithme quantique d'optimisation approximative applique en alternance un hamiltonien de problème et un hamiltonien de forçage, guidé par l'approche quantique adiabatique. Cet algorithme comporte une limitation importante : les opérateurs appliqués doivent être sous la forme d'une évolution temporelle d'un hamiltonien local fixe. Cette restriction entrave la construction d'opérateurs unitaires potentiellement plus efficaces.

L'*ansatz quantique à opérateurs alternants* (« Quantum Alternating Operator Ansatz») (QAOA), introduit par Hadfield et coll. [65], généralise l'algorithme quantique d'optimisation approximative en permettant l'alternance de familles générales d'opérateurs unitaires paramétrisés plutôt qu'uniquement des opérateurs basés sur un hamiltonien. Notons qu'un *ansatz* décrit typiquement une sous-routine composée d'une séquence de portes appliquées sur des qubits spécifiques. Cette approche supporte ainsi la représentation d'un plus grand nombre d'états, pouvant possiblement être construit de manière plus efficace. De plus, celle-ci facilite l'utilisation d'opérateur de mélange plus facilement implémentable sur le matériel informatique quantique actuel. L'algorithme quantique d'approximation quantique et l'approche quantique des opérateurs alternants possèdent le même acronyme. Comme cette dernière étend le premier, l'acronyme QAOA fera référence à l'approche quantique des opérateurs alternant pour le reste de la présente section.

Cette extension trouve son utilité principale dans la création d'opérateurs de forçage. Restreindre l'espace des configurations d'un problème selon les contraintes de celui-ci permet d'éviter une recherche de l'espace de Hilbert complet.

3.3.1 Description de l'approche

Soit un problème d'optimisation combinatoire $\varphi(x)$ et une fonction de coût $C(x)$ décrivant la qualité des entrées x de φ. Généralisant l'algorithme de Farhi et coll., QAOA est constitué de deux familles d'opérateurs : les opérateurs de séparation de phase $U_P(\gamma)$ et les opérateurs de forçage $U_D(\beta)$ où γ et β sont des paramètres réels. Notons que U_P et U_D ne sont pas restreints aux opérateurs $e^{-i\gamma H}$. Le circuit QAOA

est construit en appliquant en alternance p couches d'opérateurs des deux familles précédentes à un état initial donné.

Plutôt que de prendre en compte l'espace de Hilbert complet, un *sous-espace faisable* est considéré. Cette espace ne représente qu'un sous-ensemble de l'espace des configurations possibles, où seules les configurations respectant les contraintes du problème sont présentes. La restriction provient de la famille d'opérateurs de forçage encodant les différentes contraintes du problème φ. QAOA est donc particulièrement utile pour les problèmes comportant des contraintes rigides. Par exemple, le problème de recouvrement de sommets maximal nécessite doivent prendre la forme d'état de Dicke, c'est-à-dire une superposition de tous les états de même poids d'Hamming. Intuitivement, limiter l'espace des configurations devrait améliorer la performance de l'algorithme. Ainsi, l'état initial est généralement construit en ne considérant que les états respectant toutes les contraintes.

Quelques restrictions sont appliquées dans la conception des composantes de QAOA. D'abord, l'état initial doit être trivial à implémenter, c'est-à-dire implémentable avec un circuit de profondeur constante. Les unitaires de séparation de phases doivent être diagonales dans la base computationnelle et sont dans pris dans la majorité des cas comme $U_P(\gamma) = e^{-i\gamma H_P}$. La construction des unitaires de forçage est ici de plus grand intérêt. Ceux-ci doivent préserver le sous-espace faisable et donc transformer les états réalistes en états réalistes. Les unitaires de forçage doivent aussi fournir des transitions entre toutes les paires d'état correspondant aux états réalistes.

3.3.2 Forçage de Grover

L'approche quantique des opérateurs alternants avec forçage de Grover («Grover-Mixer Quantum Alternating Operator Ansatz») (GM-QAOA) fut proposé par Bärtschi et coll. [4] pour déplacer la complexité de la conception de l'opérateur de forçage à la préparation de l'état initial en s'inspirant de l'algorithme de Grover. Cette algorithme se base sur le cadre théorique défini dans la section précédente pour produire efficacement une superposition égale de toutes les solutions réaliste.

Décrivons d'abord le circuit QAOA, tel qu'illustré à la figure 3.4. Un opérateur

unitaire de préparation d'état U_S crée une superposition égale de toutes les solutions réalistes $|F\rangle$ dans le sous-espace faisable F. Par la suite, les unitaires de forçage U_D et de problème U_P sont appliquées en alternance p fois. Finalement, une mesure est effectuée dans la base computationnelle.

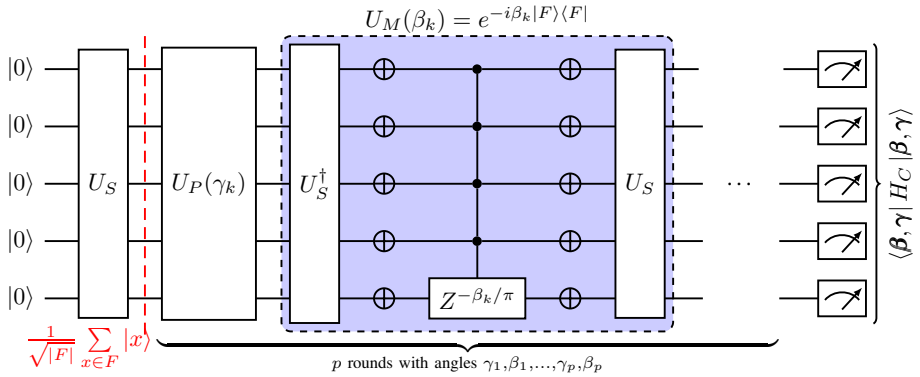

FIGURE 3.4 – *Refaire la figure.*

L'innovation principale de travail provient de l'opérateur de mélange U_D, défini comme :

$$U_D^{\text{Grover}} = e^{-i\beta|F\rangle\langle|} = U_S \left[\mathbb{1}^{\otimes n} - \left(1 - e^{-i\beta}\right)(|0\rangle\langle 0|)^{\otimes n}\right] U_S^\dagger . \tag{3.14}$$

Cet opérateur ressemble aux opérateurs de diffusion utilisés dans l'algorithme d'amplification d'amplitude, où les opérateurs de déphasage $e^{-i\beta}$ remplacent les opérateurs d'inversion de phase $e^{-i\beta}$. U_D est alors implémenté en utilisant les unitaires U_S et $U_S^\dagger$, deux couches d'opérateur de Pauli X et un opérateur de déphasage $Z^{-\beta/\pi}$ contrôlés sur tous les qubits.

Une des conséquences de l'inspiration de l'algorithme de Grover signifie GM-QAOA peut préparer des superpositions égales d'états de même énergie. De plus, comme cette approche ne possède pas d'erreur de simulation d'Hamiltonien, comme les erreurs de Trotter. Notons que les problèmes NAE3SAT et 1-in-3SAT ne sont pas contraints et donc que l'espace faisable demeure le même que pour QAOA, c'est-à-dire que $|F\rangle = |+\rangle^{\otimes n}$.

3.4 Configuration des paramètres

La capacité d'entraînement des réseaux de neurones à partir d'une simple descente de gradient fut en partie à l'origine de leur succès retentissant sur de nombreux différents problèmes. L'optimisation des paramètres se fait simplement, même avec des fonctions de coût non convexe, tout en offrant de puissants résultats. Une des attentes des VQA fut de posséder ce même comportement et de pouvoir ainsi pousser les limites des algorithmes variationnels. En effet, l'optimisation classique des paramètres de QAOA fait partie intégrante du concept d'algorithme variationnel quantique. Grâce à celle-ci, il est en théorie possible de prendre avantage des algorithmes quantiques en recourant à des ordinateurs quantiques bruités. Cependant, de nombreuses embûches rendent présentement cette tâche difficile.

D'abord, l'estimation du gradient de coût est difficile dans de nombreuses situations en raison de la présence de plateaux stériles [66, 67], où le gradient devient exponentiellement petit avec la taille du système. Un nombre exponentiel de mesures devient alors nécessaires pour pouvoir identifier la direction minimisant la fonction de coût. Ces complications se présentent surtout pour des circuits de grande profondeur, mais d'autres surgissent même pour ceux de petites tailles. Un grand nombre de minima locaux sont effectivement considérés comme pauvres, c'est-à-dire qu'ils possèdent une énergie petite par rapport au minimum global, augmentant la difficulté d'atteindre une solution approximative de bonne qualité [68]. De plus, l'optimisation des paramètres des VQA est NP-difficile, et donc intraitables dans le pire des cas [69]. Ces difficultés sont d'autant plus importantes comme la complexité de l'espace des paramètres augmente avec le nombre de paramètres d'un circuit, impliquant une plus grande difficulté d'optimisation pour les circuits profonds.

Cet ensemble de complication implique alors qu'une bonne initialisation des paramètres est nécessaire au succès de QAOA. Des paramètres initiaux suffisamment près de l'extremum global peuvent contourner les problèmes énoncés précédemment. L'initialisation de bons paramètres demeure une question ouverte, mais plusieurs approches permettent d'amoindrir ces problèmes. Sack et Serbyn propose entre autres une stratégie d'initialisation basée sur le recuit quantique trottérisé (« Trotterized Quantum Annealing ») (TQA) [70]. Cette méthode, utilisée pour les

simulations de ce travail, offre la même performance qu'un nombre exponentiel d'initialisations aléatoires. Comme vu à la section 3.2.2, un juste milieu doit être choisi entre le temps d'évolution et les erreurs de Trotter. L'initialisation TQA trouve ainsi le temps d'évolution optimale à une profondeur de circuit p fixe. Pour ce faire, la décomposition décrite à la section 3.2.2 est appliquée sur une grille uniformément discrétisée des temps d'évolution $t_k = k\Delta t$, avec $k = 1, \ldots, p$, avec un pas de temps $\Delta t = T/p$. Les angles du circuit QAOA correspondant sont alors :

$$\gamma_i = \frac{i}{p}\Delta t\,, \beta = (1 - \frac{i}{p})\Delta t\,. \tag{3.15}$$

Le pas de temps est optimisé de manière à minimiser la valeur moyenne de l'Hamiltonien de problème H_P, donnant des bons paramètres initiaux.

3.5 Échantillonnage et biais

L'allure de la distribution obtenue par QAOA est dans notre cas essentiel. Afin de se conformer à la condition de l'algorithme de JVV, celle-ci doit être quasi uniforme comme sans ce critère l'algorithme de JVV n'offre aucune assurance.

Le recuit quantique n'échantillonne pas les états fondamentaux uniformément en général [71, 72]. Certains états sont exponentiellement supprimés et nécessitent alors un nombre exponentiel de mesures pour être détectés. Comme le recuit quantique et QAOA sont fortement liés, il est ainsi peu probable que QAOA puisse échantillonner les états fondamentaux de manière uniforme.

Une propriété de l'algorithme GM-QAOA devient alors d'un intérêt considérable : l'équiprobabilité des états de même énergie. Les solutions sont encodées dans l'état fondamental de l'hamiltonien de problème, ceux-ci posséderont alors la même amplitude et donc la même probabilité. Ainsi, en ne considérant que les solutions, l'état préparé par le circuit GM-QAOA possède toujours une non-uniformité nulle.

Chapitre 4

Comptage variationnel quantique

Les chapitres précédents ont laissés transparaître l'intuition derrière l'approche empruntée dans ce travail : Est-il possible d'utiliser les algorithmes variationnels comme générateur de solutions à l'algorithme de JVV pour la résolution approximative des problèmes de comptage ? Avant d'introduire le fruit du travail de ce mémoire, l'algorithme VQCount, énumérons les difficultés liées à la résolution de problèmes de comptage avec les algorithmes variationnels quantiques. D'abord, les circuits quantiques doivent produire une grande séparation entre les amplitudes des solutions et les non-solutions de l'état produit pour pouvoir produire même une seule solution avec un nombre tractable de mesures. Ensuite, même si les amplitudes des non-solutions sont nulles, si les amplitudes d'un sous-ensemble non négligeable des solutions sont significativement supprimés par rapport aux autres amplitudes, la mesure des états supprimés entraîne une surcharge computationnelle, potentiellement exponentielle, du nombre de répétitions. Finalement, même si toutes les solutions possèdent environ des amplitudes égales et si les non-solutions ne sont pas présentes dans l'état produit, un nombre exponentiel de mesures est en théorie nécessaire pour énumérer naïvement les solutions de manière exhaustive comme le nombre de solutions est exponentiel pour les problèmes d'intérêt.

L'apport principal de ce travail réside dans la combinaison des algorithmes variationnels quantique et l'algorithme de JVV. L'algorithme VQCount échantillonne la distribution préparée par un algorithme variationnel quantique afin de produire un compte approximatif à l'aide de l'algorithme de JVV. Cette approche tente de mi-

nimiser l'impact des obstacles précédents, en particulier la deuxième et troisième difficulté. L'utilisation de GM-QAOA permet d'éviter la suppression d'amplitude d'un sous-ensemble de solutions en assurant que toutes les solutions aient la même amplitude. De plus, l'algorithme de JVV prend avantage de la structure des problèmes auto-réductibles pour éviter de devoir énumérer naïvement toutes les solutions.

Ce chapitre débute en introduisant l'algorithme VQCount à la section 4.1 et décrit la procédure d'auto-réduction nécessaire à l'algorithme VQCount à la section 4.2.

4.1 Algorithme VQCount

Comment faire le pont entre les algorithmes variationnels quantiques et l'algorithme de JVV pour la résolution de problème de comptage ? A priori, l'algorithme de JVV doit tout simplement être implémenté en utilisant un algorithme variationnel quantique comme générateur de solutions. Cependant, plusieurs embûches barrent notre chemin. Décrivons la procédure, nommé VQCount par brièveté, en affrontant ces problèmes en chemin.

Soit une instance de problème SAT, décrite par la formule CNF φ. Ce problème est auto-réductible par la relation 2.1. Ce travail se limite à l'étude du problème SAT en gardant en tête que par sa NP-complétude, une réduction existe entre n'importe quel problème de la classe NP et celui-ci. La formule φ peut être transformée en un modèle d'Ising, tel que décrit à la section 3.2.1, de l'hamiltonien du modèle d'Ising H_P encode le problème. Grâce à cet hamiltonien de problème, il est possible de construire le circuit paramétré quantique de l'algorithme QAOA, préparant l'état $|\psi(\vec{\gamma}_0, \vec{\beta}_0)\rangle$ avec les paramètres initiaux $\vec{\gamma}_0$ et $\vec{\beta}_0$. Négligeons pour le moment le choix de l'état initial et de l'hamiltonien de forçage. Une fois le circuit construit, l'énergie moyenne de H_P est évalué à l'aide de mesures répétées de l'état préparé. Un optimiseur classique modifie ensuite les paramètres du circuit quantique pour minimiser cette énergie, donnant un circuit quantique paramétré optimisé $|\psi(\vec{\gamma}, \vec{\beta})\rangle$. Les échantillons, obtenus par une mesure dans la base computationnelle de l'état préparé par le circuit, contient alors, avec une haute probabilité, des solutions à la formule φ.

QAOA étant une méthode heuristique, les paramètres optimaux ne sont pas nécessairement atteints, impliquant la présence de non-solutions dans la distribution obtenue en mesurant $|\psi(\vec{\gamma}, \vec{\beta})\rangle$. L'algorithme de JVV nécessite pourtant une distribution composée uniquement de solutions, impliquant qu'une étape de post-traitement doit être employé pour retirer les non-solutions. Comme le problème SAT appartient à la classe NP, vérifier qu'un échantillon est une solution est un processus efficace. Ainsi, toutes les non-solutions échantillonnées sont retirées de la distribution obtenue. De plus, comme discuté à la section 3.5, la non-uniformité de la distribution obtenue ne respecte pas nécessairement la condition de l'algorithme de JVV. La discussion dans cette section se restreindra alors à la variante GM-QAOA, qui garantie que les amplitudes des solutions préparées soient égales. Ainsi, le critère de mérite pertinent est la probabilité que le résultat d'une mesure soit une solution, ou plus succinctement le *taux de succès* :

$$r = \langle \psi(\vec{\gamma}, \vec{\beta}) | \hat{\mathcal{P}}_G | \psi(\vec{\gamma}, \vec{\beta}) \rangle \tag{4.1}$$

où $\hat{\mathcal{P}}_G$ est le projecteur dans l'espace des solutions G, c'est-à-dire l'état fondamental du modèle d'Ising représentant le problème, donné par

$$\hat{\mathcal{P}}_G = \sum_{x \in G} |x\rangle\langle x| \tag{4.2}$$

Cette définition permet alors de décrire la performance des différents algorithmes pour le comptage basé sur l'échantillonnage sur un pied d'égalité en utilisant le taux de succès r, la tolérance (ou l'erreur multiplicative) ε ainsi que la confiance δ. Cette caractérisation s'étend aux algorithmes classiques employant un générateur échouant à produire une solution à un taux $1 - r$.

Ces remarques complétées, des échantillons sont mesurés à partir du circuit $|\vec{\gamma}, \vec{\beta}\rangle$ pour estimer la valeur la plus probable de la première variable w_0. Selon l'algorithme de JVV, il faut alors échantillonner des solutions à la sous-instance φ_{w_0}. Résoudre directement celle-ci à l'aide du même circuit optimisé demanderait une post-sélection supplémentaire sur la première variable. En réalité, comme il existe un nombre exponentiel de sous-instances en raison du nombre exponentiel de chaînes

de bits possibles, un nombre exponentiel d'échantillons supplémentaires devraient être retiré. Une autre méthode consiste à construire et optimiser un différent circuit pour résoudre la sous-instance. Pour éviter cette surcharge computationnelle, le circuit de GM-QAOA est modifié pour résoudre directement la sous-instance. Cette procédure d'auto-réduction, qui consiste à remplacer pour chaque qubit fixé la porte d'Hadamard par une porte de Pauli conditionnée sur la valeur w_0 du bit fixé et à retirer l'opérateur de l'hamiltonien de forçage sur le qubit associé, est détaillée à la section 4.2. Ce changement permet de conserver la propriété d'égalité des amplitudes de GM-QAOA. De plus, celle-ci suggère qu'il soit possible d'éviter d'optimiser à nouveau le circuit en gardant les mêmes paramètres. Ainsi, le circuit est modifié selon la procédure d'auto-réduction sans optimiser les paramètres. Des solutions sont alors encore échantillonnées de ce sous-circuit et le processus est répété de manière récursive jusqu'à ce que toutes les variables soient fixées. Le nombre de solutions est alors données par l'inverse du produit des probabilités conditionnelles tel qu'énoncé par l'algorithme de JVV. L'algorithme 1 résume les étapes de l'algorithme VQCount.

Comme l'algorithme VQCount se fonde sur l'algorithme de JVV, son utilisation requiert $O(\frac{n^2 \log(\frac{1}{\delta})}{r\varepsilon^2})$ échantillons pour approximer le nombre d'états fondamentaux à une erreur multiplicative ε avec une confiance $1 - \delta$. Ce résultat se compare favorablement à un travail similaire évaluant la fonction de partition d'hamiltoniens de spin classiques qui requiert $O(\frac{\sqrt{N \log(1/\delta)}}{r\varepsilon})$ échantillons pour la même tâche [73]. Lorsque $N = O(2^n)$, VQCount nécessite exponentiellement moins d'échantillons que cette méthode.

Malgré que l'approche présentée résoute certains des problèmes énoncés au début de la section, les problèmes #P-difficile demeure difficile pour les ordinateurs quantiques. La génération des distributions arbitraires, incluant les distributions uniformes de structures combinatoires, est difficile en général et seulement facile dans des cas exceptionnels [13, 14]. De plus, GM-QAOA requiert en général des profondeurs de circuits de taille exponentielle selon la taille du problème pour atteindre un taux de succès fini [74].

Notons qu'un algorithme classique peut être utilisé pour résoudre l'instance du problème de comptage lorsque VQCount ait suffisamment réduit celle-ci.

Algorithm 2: VQCount

Require: Nombre de variables : n, Hamiltonien de problème : H_P, Hamiltonien de forçage : H_D, Paramètres initiaux : $(\vec{\beta}_0, \vec{\gamma}_0)$, Profondeur : p, Nombre d'étapes d'optimisation : n_o, Nombre de solutions à échantillonner : n_s

PQC$(\vec{\beta}_0, \vec{\gamma}_0) \leftarrow$ Circuit-QAOA$(H_P, H_D, \vec{\beta}_0, \vec{\gamma}_0, p)$
PQC$(\vec{\beta}, \vec{\gamma}) \leftarrow$ Optimisation(PQC$(\vec{\beta}_0, \vec{\gamma}_0), n_o)$
$w \leftarrow$ ""
$\tilde{N} \leftarrow 1$
for $i \in \{1, \ldots, n\}$ **do**
  $S \leftarrow \{\ \}$
  while $|S| < n_s$ **do**
    $m \leftarrow$ Mesure(PQC$(\vec{\beta}, \vec{\gamma}))$
    if Vérification-Solution(m) **then**
      $S \leftarrow S \cup \{m\}$
    end if
  end while
  $w, \tilde{p} \leftarrow$ Préfixe-Majoritaire(S)
  PQC$(\vec{\beta}, \vec{\gamma}) \leftarrow$ Auto-Réduction(PQC$(\vec{\beta}, \vec{\gamma}), w)$
  $\tilde{N} \leftarrow \tilde{N}/\tilde{p}$
end for
return $\tilde{N}$

4.2 Procédure d'auto-réduction

Pour agir en tant que générateur de solutions à sous-instance de l'instance du problème original, le circuit GM-QAOA doit être modifié pour être en mesure d'échantillonner à partir de la distribution de solutions appropriée. Comme expliqué à la section précédente, il n'est pas suffisant d'échantillonner l'état produit par le circuit original en raison du nombre exponentielle de mesures nécessaires. Une *procédure d'auto-réduction*, illustrée à la figure 4.1, est ainsi introduite pour résoudre cette complication. Soit une formule CNF φ et une sous-formule CNF φ_w représentant la formule φ où les premières variables sont remplacées par w. Alors, pour chaque qubit q fixé à w_q du circuit GM-QAOA, les opérations suivantes sont appliquées :

(1) Retirer la porte d'Hadamard H initiale du qubit q.

(2) Insérer un porte de Pauli X conditionnée par la valeur w_q.

(3) Retirer l'opérateur de l'hamiltonien de forçage H_D du qubit q.

Ces modifications fixent alors l'état du qubit q à $|w_q\rangle$ tout en préservant les termes d'interactions encodant les contraintes de la formule φ_w. De plus, le circuit modifié possède les propriétés de GM-QAOA. L'annexe A prouve la validité de ces affirmations.

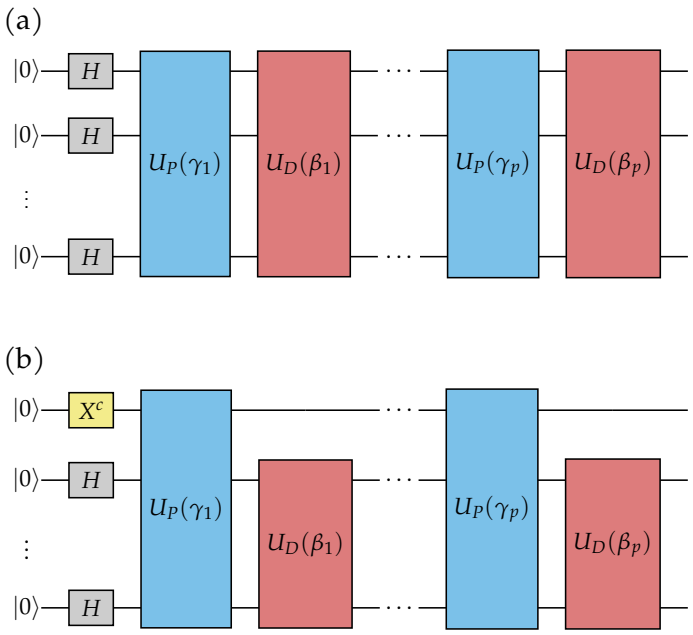

FIGURE 4.1 – Générateur de solutions QAOA avant (a) et après (b) avoir fixé le premier qubit à c durant la procédure d'auto-réduction.

L'algorithme VQCount ne comprend qu'une optimisation des paramètres du circuit quantique paramétré avant la procédure d'auto-réduction. Toutefois, un autre cycle d'optimisation est en principe nécessaire pour chaque circuit réduit afin d'atteindre un état optimal représentant les solutions à la sous-instance du problème. Ce choix mène alors à la question suivante : Est-ce que la probabilité de succès r du circuit réduit est égale ou supérieur à celle du circuit original avec les mêmes paramètres ? Dans ce cas, l'optimisation n'est nécessaire que pour le circuit original et peut être sautée pour les circuits réduits.

La réponse à cette question est positive dans le cas de circuits GM-QAOA peu profonds dans la limite $\gamma_i = \beta_i, i = 1, \ldots, p$. Dans cette limite, GM-QAOA se réduit à

l'algorithme de Grover. En effet, pour tous les hamiltoniens de problème H_P pouvant être écrit comme une somme de chaînes de Pauli, un circuit peut être construit effondrant tous les niveaux excités de H_P à un seul niveau excité, menant à circuit d'évolution d'hamiltonien à deux niveaux U_P. Avec les paramètres susmentionnés, U_P devient un oracle pour les états fondamentaux de H_P, à une phase constante près, et U_D devient l'opérateur de diffusion de Grover. Dans cette limite, à une profondeur constante $p = O(1)$, le taux de succès est une fonction strictement croissante du rapport des solutions au nombre total d'états. Si, en descendant l'arbre d'auto-réductibilité de l'algorithme de JVV, nous choisissons les branches possédant une probabilité plus grande que $1/2$, alors le ratio des solutions au nombre total d'états est aussi strictement croissant. Cela signifie que, pour ce choix de paramètres, un circuit GM-QAOA peu profond pour un problème donné peut être modifié selon la procédure d'auto-réduction pour résoudre les sous-problèmes avec le même ou un meilleur taux de succès. Cet argument est seulement valide dans la limite de Grover, mais suggère que la stratégie d'auto-réduction est possiblement aussi valide dans d'autres cas. Le chapitre suivant introduit des évidences sous la forme de simulations numériques pour confirmer la validité de cette intuition.

Chapitre 5

Résolution de problèmes #P-complet avec VQCount

Dans la section précédente, un cadre théorique s'appuyant sur l'algorithme de JVV a été construit pour la résolution de problèmes de comptage à l'aide d'algorithmes variationnels quantiques. L'algorithme résultant, VQCount, obtient un compte approximatif à une erreur multiplicative près en échantillonnant un nombre polynomial de solutions à un circuit quantique optimisé GM-QAOA. Toutefois, plusieurs conditions potentiellement dispensables ou difficilement applicables ont été imposées. Cette section change l'optique précédente en évaluant l'algorithme VQCount uniquement en tant que méthode heuristique, c'est-à-dire en caractérisant l'algorithme sans se soucier des garanties. La résolution de deux problèmes #P-complet, #NAE3SAT et #1in3SAT, sont étudiés.

La section 5.1 définit les paramètres des simulations numériques effectuées et la section 5.2 explique la méthode utilisée pour caractériser l'algorithme VQCount. La performance de VQCount est examinée aux sections 5.3 et 5.4, où l'impact du biais d'échantillonnage des circuits QAOA et le comportement d'échelles sont respectivement présentés.

Le projet de ce mémoire a mené au développement d'un module Python, nommé VQCount, disponible sur GitHub (`https://github.com/QuICoPhy-Lab/VQCount`). Ce module a été utilisé pour les simulations numériques de ce chapitre.

5.1 Paramètres de l'étude

Certaines précédentes considérations sont relâchées ou modifiées. D'abord, l'algorithme GM-QAOA n'est plus considéré dans sa forme limite de Grover. Plutôt qu'utiliser les paramètres du circuit fixes $\gamma_i = \beta_i = \pi$ pour $i = 1, \ldots, p$, les paramètres sont optimisés. De plus, les problèmes étudiés sont transformés à des hamiltoniens de problème régulier contenant une tour d'états excités plutôt que de l'oracle à deux niveaux construit précédemment. Comme l'impact de la non-uniformité de la distribution d'état produite, l'algorithme VQCount est aussi employé avec l'algorithme quantique d'optimisation approximative.

Pour les simulations numériques effectuées, les solutions sont échantillonnées sans remplacement. Ce choix dévie du protocole de l'algorithme de JVV, mais améliore l'efficacité de la méthode lorsqu'un problème possède peu de solutions ou lorsque la distribution préparée par le circuit est fortement non-uniforme.

L'algorithme VQCount est appliqué aux problèmes #NAE3SAT positif et #1-in-3SAT positif présentés à la section 1.2. Le problème #NAE3SAT est difficile près du seuil de satisfaisabilité situé au ratio de clauses sur variables critique $\alpha_c \approx 2.1$. Pour étudier l'effet de la complexité des formules aléatoires sur la performance de VQCount, des instances sont générées à $\alpha = 1$ et $\alpha = 2$. Celles-ci sont générées à partir de graphes connectés bipartis, imposant une restriction supplémentaire. Similairement, les instances du problème #1-in-3SAT positive sont à leur seuil de difficulté le plus élevé près de $\alpha_c \approx 2/3$. Pour étudier ce problème dans le régime difficile, les instances sont générées à partir de graphes aléatoires cubiques, en plaçant une clause sur tous les sommets et une variable sur toutes les arrêtes. Cette méthode permet d'échantillonner uniformément les instances difficiles, éliminant le biais dû à une sélection aléatoire [75]. Ce type de problème appartient à la catégorie des problèmes verrouillés, décrit à la section 1.5, étant vraisemblablement les plus difficiles des problèmes #P. Par la suite, les instances générées sont converties en un modèle d'Ising à l'aide de la transformation décrite à la section 3.2.1.

La simulation de circuits quantiques est un problème complexe en raison de la quantité de mémoire nécessaire pour représenter l'espace de Hilbert. La méthode la plus utilisée, la simulation à vecteur d'état, ne peut atteindre plus d'une vingtaine

de qubits pour cette raison. Les réseaux de tenseurs sont un outil alternatif puissant, particulièrement pour l'étude de circuits quantiques peu profonds. L'annexe B introduit les réseaux de tenseur et les méthodes particulières utilisées pour la simulation de l'algorithme VQCount. La librairie « Quimb » [76] est utilisé pour modéliser l'optimisation et l'échantillonnage des circuits quantiques QAOA et GM-QAOA. Les paramètres de ces circuits sont optimisés avec la programmation séquentielle des moindres carrés telle qu'implémenté par la librairie « SciPy » [77]. Pour évaluer la validité des résultats obtenus, « Ganak » [78], un compteur de modèles exact basé sur une approche probabiliste, est utilisé pour trouver le nombre exact de solutions et le solveur SAT « Glucose3 » [79, 80], implémenté dans l'ensemble d'outils « PySAT » [81], est utilisé pour énumérer toutes les solutions.

Pour le problème #NAE3SAT, 20 instances aléatoires sont générées par taille d'instance, allant de 6 à 20 variables, pour les deux densités de clause $\alpha = 1$ et $\alpha = 2$. Similairement, problème #1-in-3SAT, 20 instances aléatoires sont générées pour une taille de variable allant de 9 à 27 variables par saut de 3 variables, pour une densité de clause $\alpha = 2/3$.

5.2 Méthode

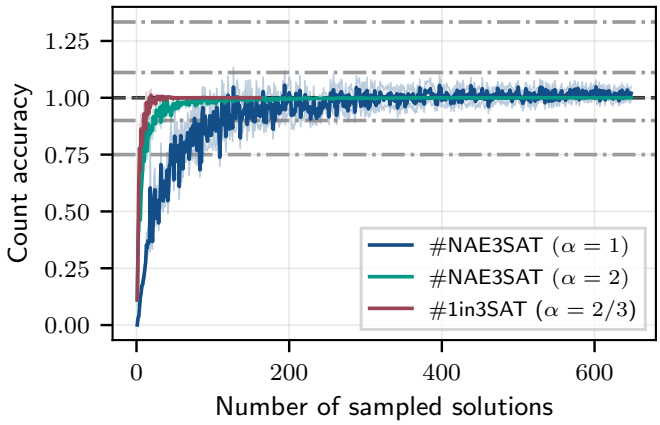

FIGURE 5.1 – Précision du comptage obtenue en augmentant le nombre de solutions échantillonnées à chaque étape de l'auto-réduction à partir du circuit QAOA de profondeur $p = 3$ pour des instances #NAE3SAT et #1-in-3SAT de $n = 18$ variables. Les lignes grises en pointillés et en tirets représentent les erreurs multiplicatives $\varepsilon = \frac{1}{3}$ et $\varepsilon = \frac{1}{9}$.

Afin de déterminer le nombre d'échantillons nécessaires pour atteindre un compte approximatif avec une tolérance ε et une confiance δ, le nombre d'échantillons utilisé doit être augmenté par l'algorithme VQCount jusqu'à ce que le nombre de solutions trouvé soit à une erreur multiplicative ε du nombre de solutions exact avec une probabilité supérieure à δ. La simulation de l'algorithme étant coûteuse, une méthode plus simple et moins rigoureuse est utilisée ici. L'algorithme VQCount est exécuté en augmentant le nombre de solutions échantillonnées du circuit quantique optimisé QAOA jusqu'à ce que le nombre de solutions soit à l'intérieur des bornes d'erreur ε. La figure 5.1 montre la précision du comptage, c'est-à-dire le nombre de solutions approximatif sur le nombre de solutions exact, pour différentes erreurs multiplicatives ε. Notons que pour réduire les incertitudes, une procédure de lissage est utilisée. L'algorithme de JVV semble sous-estimer le nombre de solutions, comme vu avec la courbe de #NAE3SAT avec $\alpha = 1$. Ce comportement s'explique par le manque d'échantillons indiscernables dans la distribution de probabilité des solutions échantillonnées, entraînant une légère inexactitude des probabilités dans l'arbre des configurations.

5.3 Biais d'échantillonnage

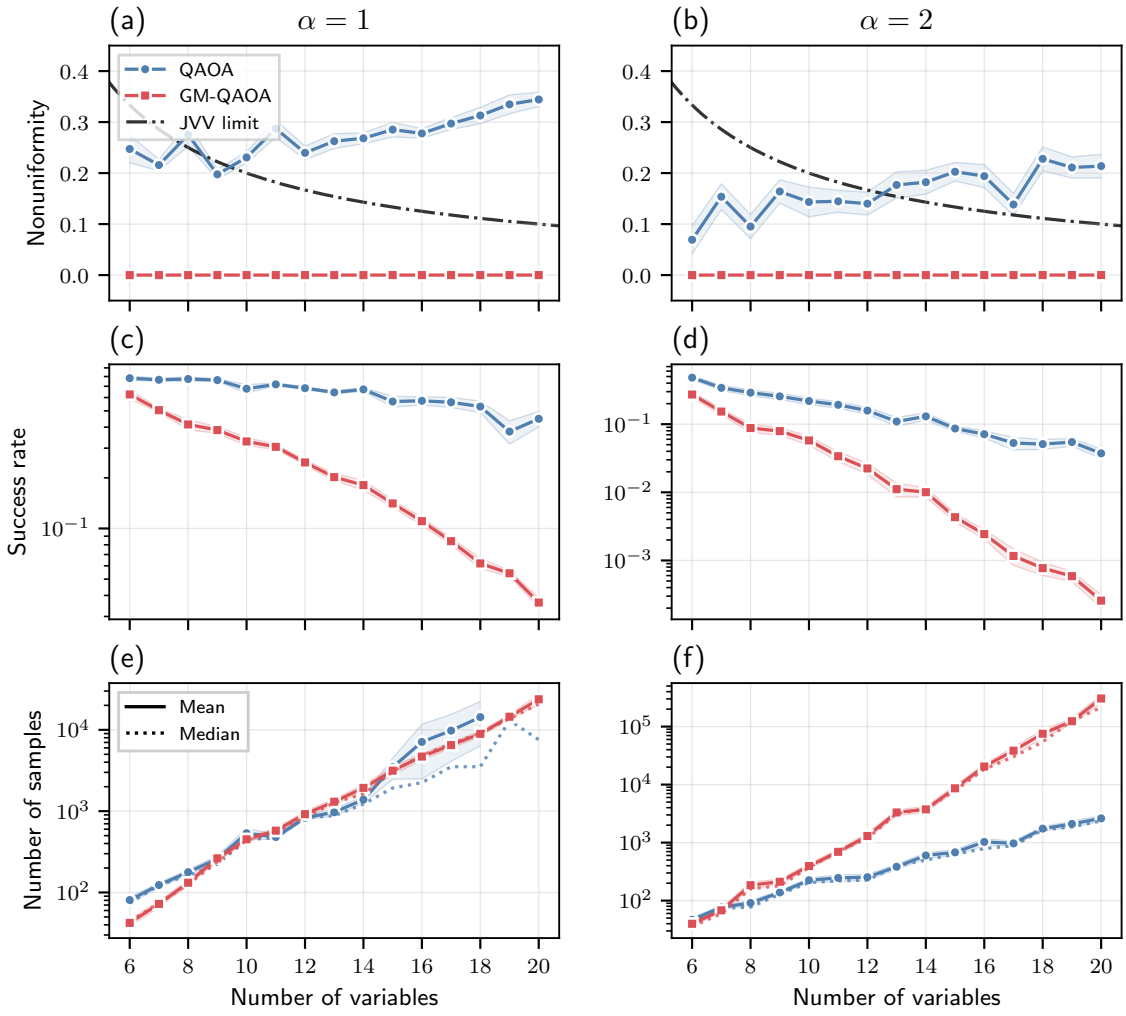

FIGURE 5.2 – Comparaison de la performance de l'algorithme VQCount avec des circuits QAOA et GM-QAOA de profondeur $p = 3$ pour des instances NAE3SAT à $\alpha = 1$ (gauche) et à $\alpha = 2$ (droite). (a, b) Non-uniformité maximale à travers l'auto-réduction. La limite de JVV montrée est donnée par $f(n) = \frac{2}{n}$. (c, d) Taux de succès minimum à travers l'auto-réduction. (e, f) Moyenne (lignes solides) et médiane (lignes pointillées) du nombre d'échantillons nécessaires pour obtenir un compte approximatif avec une tolérance $\varepsilon = \frac{1}{3}$. Un ajustement exponentiel de la courbe extrapolé des 5 derniers points de la médiane donne $O(1.45^n)$ (QAOA) et $O(1.44^n)$ (GM-QAOA) pour (e) ainsi que $O(1.34^n)$ (QAOA) et $O(1.88^n)$ (GM-QAOA) pour (f).

Comme vu à la section 3.5, l'état préparé par un circuit GM-QAOA permet un échantillonnage uniforme de solutions tel que nécessaire pour l'algorithme de JVV, alors que la probabilité de certaines solutions est exponentiellement petite avec l'approche QAOA. Sachant que les problèmes étudiés possèdent en moyenne un nombre exponentiel de solutions et qu'un nombre polynomial d'échantillons est nécessaire, il est possible que la distribution préparée par QAOA soit suffisante pour obtenir un compte approximatif convenable. La figure 5.2 résume les résultats obtenus pour le problème #NAE3SAT avec un circuit de profondeur $p = 3$. Les figures 5.2(a) et 5.2(b) confirme que GM-QAOA échantillonne les solutions parfaitement uniformément. Au contraire, la distribution des solutions préparées par QAOA est de plus en plus non-uniforme avec la taille de l'instance. Les figures 5.2(c) et 5.2(d) montrent que, pour une profondeur de circuit fixe, le taux de succès de QAOA et GM-QAOA diminue grossièrement de façon exponentielle avec le nombre de variables, malgré que la diminution est plus importante pour GM-QAOA. Les taux de succès sont plus faibles et se détériorent plus rapidement pour $\alpha = 2$ que pour $\alpha = 1$, reflétant la différence de complexité du problème selon la proximité du seuil de satisfaisabilité. Le nombre total d'échantillons requis pour que l'algorithme VQCount retourne un compte approximatif à une erreur multiplicative $\varepsilon = 1/3$ est présenté à la figure 5.2(e) et 5.2(f). Le nombre d'échantillons augmente exponentiellement avec la taille des instances en raison de la post-sélection des solutions à la fois pour QAOA et GM-QAOA comme générateurs de solutions. Malgré le taux de succès exponentiellement plus faible de GM-QAOA, VQCount nécessite environ le même nombre d'échantillons avec QAOA ou GM-QAOA pour $\alpha = 1$. Au contraire, VQCount avec QAOA requiert un nombre exponentiellement plus faible qu'avec GM-QAOA, malgré la non-uniformité de la distribution des probabilités de QAOA. Cela suggère que QAOA est potentiellement un générateur plus efficace pour VQCount dans le cas où les solutions soient rares et dispersées. Le nombre d'échantillons post-sélecté nécessaire pour que VQCount produise un estimé à un facteur multiplicatif ε près du compte exact, présenté à la figure 5.3, semble polynomial selon le nombre de variables n et l'inverse de la tolérance $1/\varepsilon$ comme attendu.

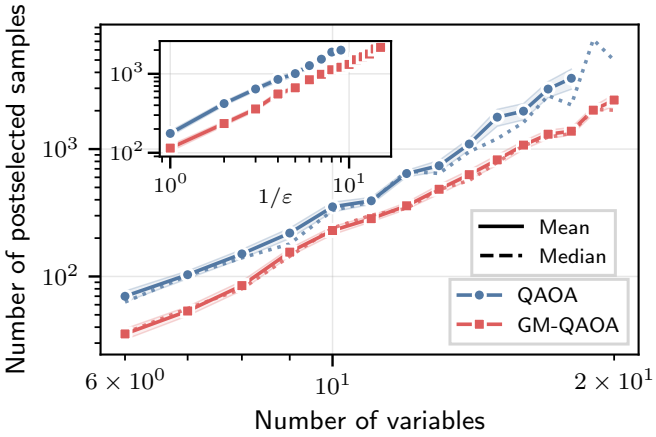

FIGURE 5.3 – Nombre d'échantillons post-sélectionnés nécessaires pour obtenir un comptage approximatif avec une tolérance d'erreur $\varepsilon = 1/3$, en utilisant des circuits QAOA et GM-QAOA de profondeur $p = 3$ pour des instances de NAE3SAT à $\alpha = 1$. Un ajustement polynomial de la médiane donne $O(n^{5.12})$ (QAOA) et $O(n^{3.42})$ (GM-QAOA). Le panneau en encart montre le comportement de mise à l'échelle en fonction de l'inverse de la tolérance d'erreur ε pour $n = 12$ variables. Un ajustement polynomial de la moyenne donne $O(\varepsilon^{-1.09})$ (QAOA) et $O(\varepsilon^{-1.07})$ (GM-QAOA).

Dans cette étude, les paramètres des circuits quantiques sont maintenus à une profondeur fixe lors de la procédure de JVV comme la ré-optimisation est coûteuse à simuler classiquement. Est-ce que ce choix a un impact important sur la performance de l'algorithme VQCount ? Sachant qu'à chaque étape de l'algorithme de JVV un problème plus petit est échantillonné, il est raisonnable de s'attendre à ce que le succès de l'algorithme augmente. Comme illustré à la figure 5.4, la non-uniformité diminue en moyenne lorsque des qubits additionnels sont fixés pour GM-QAOA.

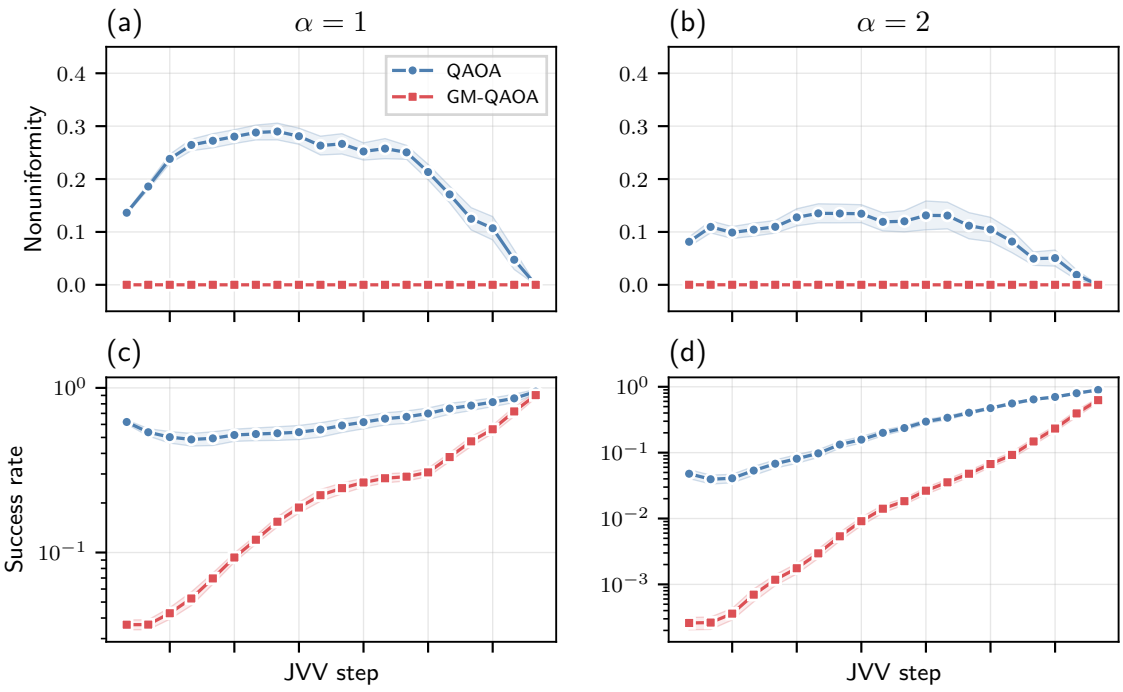

FIGURE 5.4 – Impact de la fixation de qubits au long de l'auto-réduction sur l'échantillonnage des circuits QAOA et GM-QAOA de profondeur $p = 3$ pour des instances de NAE3SAT avec $n = 18$ variables à $\alpha = 1$ (panneaux de gauche) et $\alpha = 2$ (panneaux de droite). (a, b) Non-uniformité en fonction du nombre d'étapes de l'auto-réduction. (c, d) Taux de succès en fonction du nombre d'étapes de l'auto-réduction.

Pour l'instant, la performance de VQCount a été étudié pour des circuits de profondeur fixe. La figure 5.5 montre que, lorsque la profondeur du circuit augmente, la non-uniformité diminue et la probabilité d'obtenir une solution augmente, menant à un nombre d'échantillons nécessaires pour atteindre un compte approximatif avec une erreur multiplicative donnée. QAOA semble s'améliorer plus rapidement que GM-QAOA, mais la non-uniformité de QAOA augmente et atteint un plateau avec la profondeur du circuit.

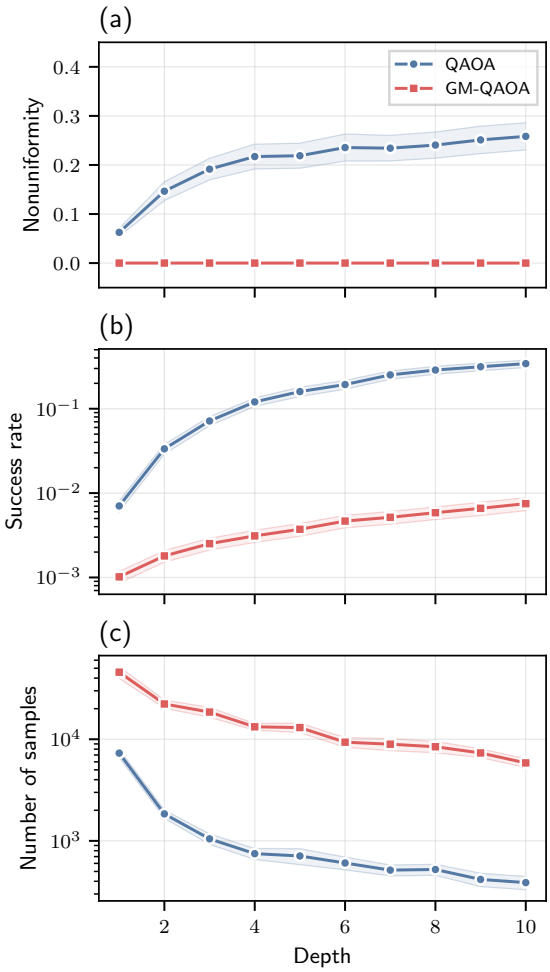

FIGURE 5.5 – Performance de l'algorithme VQCount en fonction de la profondeur avec QAOA et GM-QAOA pour des instances de NAE3SAT avec 16 qubits à $\alpha = 2$. (a) Non-uniformité maximale au cours de l'auto-réduction. (b) Taux de succès minimal au cours de l'auto-réduction. (c) Nombre moyen d'échantillons nécessaires pour que le comptage approximatif soit dans une tolérance d'erreur $\varepsilon = \frac{1}{3}$.

Les résultats précédents suggèrent que le circuit QAOA est un meilleur générateur de solutions que le circuit GM-QAOA pour une faible profondeur. GM-QAOA dépend aussi d'une porte contrôle à n qubits, difficilement implémentable pour la simulation classique et pour les ordinateurs quantiques bruités à court-terme.

5.4 Performance

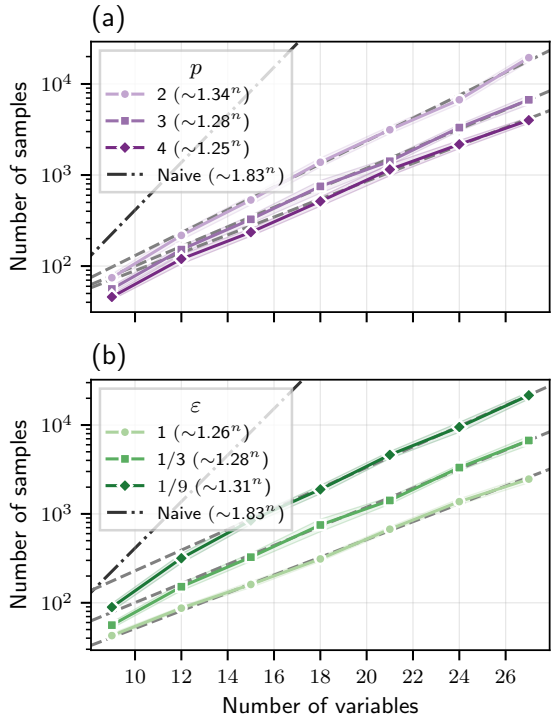

FIGURE 5.6 – Comportement d'échelle de l'algorithme VQCount avec QAOA pour des instances de 1-in-3SAT à $\alpha = \frac{2}{3}$. L'échantillonnage par rejet naïf est représenté par une ligne noir pointillé-mixte. Les lignes en pointillés indiquent l'ajustement exponentiel extrapolé à partir des 4 derniers points. (a) Nombre d'échantillons nécessaires pour que le comptage approximatif soit dans une tolérance d'erreur $\varepsilon = \frac{1}{3}$ pour différentes profondeurs. (b) Nombre d'échantillons nécessaires pour que le comptage approximatif soit dans une tolérance d'erreur ε pour une profondeur $p = 3$.

Ayant conclu que QAOA semble être un meilleur générateur de solutions pour une faible profondeur de circuit, la performance de VQCount avec QAOA est caractérisé de manière approfondie pour le problème #1-in-3SAT. La figure 5.6 résume le comportement de mise à l'échelle de VQCount avec un circuit QAOA à faible profondeur. Naïvement, le nombre de solutions peut être trouvé en déterminant la probabilité qu'une chaîne de bit aléatoire échantillons de l'ensemble des chaînes de bits possible soit une solution. Comme vu à la figure 5.6(a), VQCount possède une

loi d'échelle exponentiellement meilleure qu'avec un échantillonnage par rejet naïf et sa performance s'améliore avec la taille du circuit. Aux profondeurs de circuit possibles d'atteindre en utilisant des ressources computationnelles raisonnables, la performance asymptotique de VQCount est plus faible que les solveurs classiques de pointe pour ce problème [34], mais les résultats suggèrent que ce n'est possiblement pas le cas pour des profondeurs de circuit finies plus grandes. La précision de VQCount peut être améliorée en augmentant le nombre d'échantillons, comme montré à la figure 5.6(b). Finalement, la figure 5.7 montre le nombre d'échantillons post-sélectés nécessaire pour produire un compte à un facteur multiplicatif ε près du nombre de solutions exact en fonction du nombre de variables et de l'inverse de la tolérance. Ces résultats suggèrent un comportement d'échelle superpolynomiale, mais sous-exponentiel du nombre d'échantillons post-sélectés nécessaire.

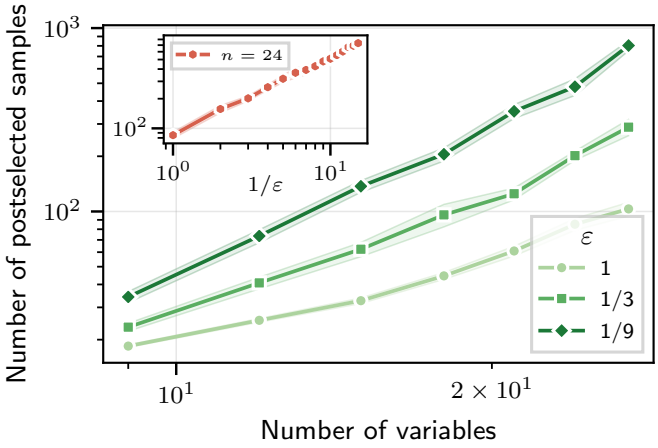

FIGURE 5.7 – Nombre d'échantillons post-sélectionnés nécessaires pour obtenir un comptage approximatif avec une tolérance d'erreur ε pour des circuits QAOA de profondeur $p = 3$ appliqués à des instances de 1-in-3SAT à $\alpha = \frac{2}{3}$. Le panneau en encart montre le comportement de mise à l'échelle en fonction de l'inverse de la tolérance d'erreur ε pour $n = 24$ variables, pour lequel un ajustement polynomial donne $O(\varepsilon^{-0.79})$.

L'algorithme VQCount est conçu pour estimer un nombre exponentiel de solutions en utilisant seulement un nombre polynomial d'échantillons. Toutefois, dans les simulations numériques, certaines instances de problème peuvent ne comporter que peu de solutions. Il est donc essentiel de vérifier que VQCount requiert, en pratique, moins d'échantillons que le nombre total de solutions, afin d'éviter un

simple énumération exhaustive. La Fig. 5.8 illustre l'efficacité d'échantillonnage de VQCount, définie comme le rapport entre le nombre exact de solutions et le nombre de solutions distinctes utilisées par l'algorithme, pour les deux problèmes étudiés dans ce travail. On observe que l'efficacité d'échantillonnage est positivement corrélée à la densité des solutions. Néanmoins, VQCount utilise systématiquement moins d'échantillons que le nombre total de solutions.

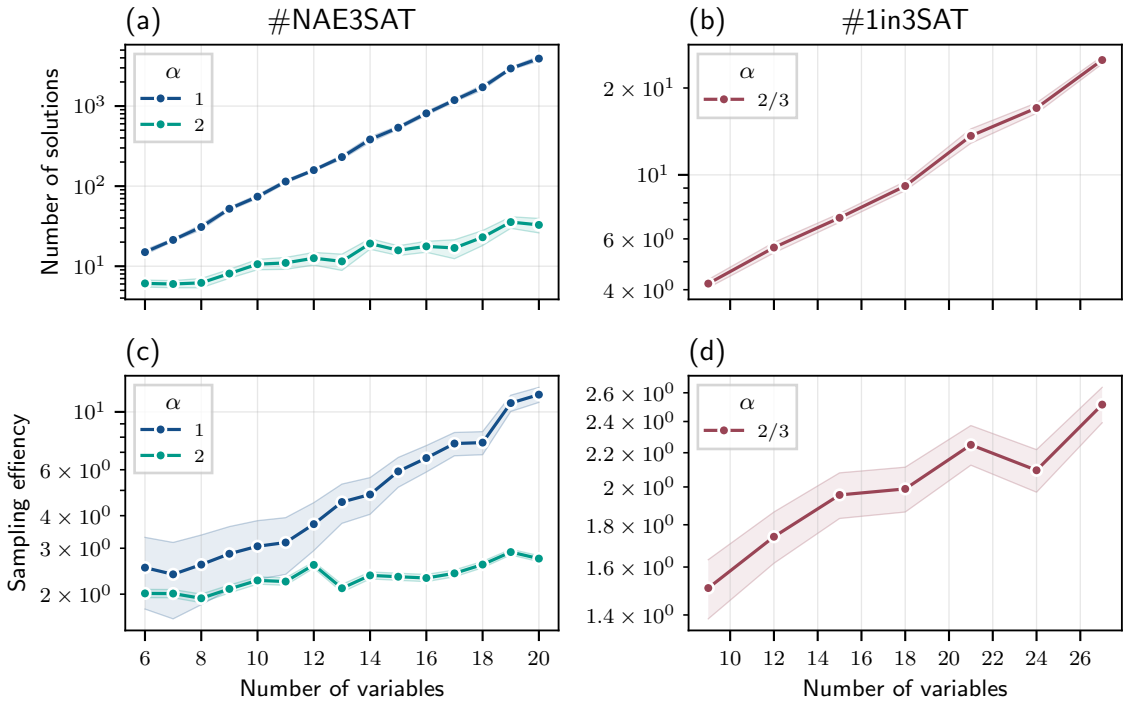

FIGURE 5.8 – Nombre de solutions (a) et efficacité d'échantillonnage (b) nécessaires pour atteindre une tolérance d'erreur $\varepsilon = 1$ avec l'algorithme VQCount utilisant QAOA de profondeur $p = 3$ pour des instances de NAE3SAT et 1-in-3SAT.

Conclusion

Annexe A

Auto-réductibilité du circuit GM-QAOA

Théorème A.1 – Auto-réductibilité de GM-QAOA

Supposons qu'un circuit GM-QAOA génère efficacement des solutions approximatives à un problème auto-réductible φ dans #P. Alors, ce circuit peut être modifié, sans modifier les paramètres du circuit, pour résoudre approximativement tous les sous-problèmes φ_c de φ en conservant les propriétés de GM-QAOA.

Démonstration. Soit φ l'instance d'un problème de #P de taille n que nous voulons résoudre avec GM-QAOA. Cette instance peut être encodée dans un hamiltonien de problème H_P avec des valeurs propres $g = 0$ pour les solutions et $e^{(k)} = \varepsilon_k \in \mathbb{R}_{>0}$ pour les non-solutions. L'opérateur U_P correspondant avec H_P s'écrit alors comme :

$$U_P = \sum_{j \in G} |j\rangle \langle j| + \sum_k e^{-i\gamma\varepsilon_k} \sum_{j \in E^{(k)}} |j\rangle \langle j| \,, \tag{A.1}$$

où G et $E^{(k)}$ sont respectivement les variétés des états fondamentaux et états excités, et donc l'ensemble des solutions et des non-solutions de φ. Pour une espace de solutions non contraint, l'opérateur U_D^{GM} est donné par :

$$U_D^{GM} = \mathbb{1} - (1 - e^{-i\beta})(|+\rangle\langle+|^{\otimes n}) \,. \tag{A.2}$$

Soit φ_c le sous-problème de φ où les dernières q variables sont fixées par la chaîne de bits c. Définissons G_c et $E_c^{(k)}$ comme les ensembles de solutions and non-solutions

de φ_c. Les états fondamentaux et excités s'écrivent comme :

$$|g_c\rangle = \frac{1}{\sqrt{|G_c|}} \sum_{j \in G_c} |j\rangle \,, \tag{A.3}$$

$$|e_c^{(k)}\rangle = \frac{1}{\sqrt{|E_c^{(k)}|}} \sum_{j \in E_c^{(k)}} |j\rangle \,. \tag{A.4}$$

Évoluons l'état initial $|\psi_0\rangle = |+\rangle^{\otimes n}$ avec un circuit optimisé GM-QAOA de p couches en séparant le registre des derniers q qubits. Montrons par induction que l'état final suit la formule récursive suivante :

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{Q-1} \left(\sqrt{|G_i|} F_p \, |g_i\rangle + \sum_k \sqrt{|E_i^{(k)}|} \bar{F}_p^{(k)} \, |e_i^{(k)}\rangle \right) |i\rangle \,, \tag{A.5}$$

où $N = 2^n$, $Q = 2^q$, $F_0 = \bar{F}_0^{(k)} = 1$ et

$$F_p = F_{p-1} - \frac{1}{N}(1 - e^{-i\beta_p}) \left(|G| F_{p-1} + \sum_{k'} |E^{(k')}| \bar{F}_{p-1}^{(k')} e^{-i\gamma_p \varepsilon_{k'}} \right) \,, \tag{A.6}$$

$$\bar{F}_p^{(k)} = e^{-i\gamma_p \varepsilon_k} \bar{F}_{p-1}^{(k)} - \frac{1}{N}(1 - e^{-i\beta_p}) \left(|G| F_{p-1} + \sum_{k'} |E^{(k')}| \bar{F}_{p-1}^{(k')} e^{-i\gamma_p \varepsilon_{k'}} \right) \,. \tag{A.7}$$

Cas 1 (1 couche) :

Appliquons U_P en calculant $|\psi_1\rangle = U_P\,|\psi_0\rangle$:

$$
\begin{aligned}
|\psi_1\rangle &= \left(\sum_{j\in G}|j\rangle\langle j| + \sum_k e^{-i\gamma_1\varepsilon_k}\sum_{j\in E^{(k)}}|j\rangle\langle j|\right)\left(\frac{1}{\sqrt{\frac{N}{Q}}}\sum_{j'=0}^{N/Q-1}|j'\rangle\right)\left(\frac{1}{\sqrt{Q}}\sum_{i'=0}^{Q-1}|i'\rangle\right)\\
&= \left(\sum_{i=0}^{Q-1}\sum_{j\in G_i}|j\rangle\langle j|\otimes|i\rangle\langle i| + \sum_{i=0}^{Q-1}\sum_k e^{-i\gamma_1\varepsilon_k}\sum_{j\in E_i^{(k)}}|j\rangle\langle j|\otimes|i\rangle\langle i|\right)\left(\frac{1}{\sqrt{N}}\sum_{i'=0}^{Q-1}\sum_{j'=0}^{N/Q-1}|j'\rangle\otimes|i'\rangle\right)\\
&= \frac{1}{\sqrt{N}}\left(\sum_{i=0}^{Q-1}\sum_{j\in G_i}|j\rangle\otimes|i\rangle + \sum_{i=0}^{Q-1}\sum_k e^{-i\gamma_1\varepsilon_k}\sum_{j\in E_i^{(k)}}|j\rangle\otimes|i\rangle\right)\\
&= \frac{1}{\sqrt{N}}\sum_{i=0}^{Q-1}\left(\sum_{j\in G_i}|j\rangle + \sum_k e^{-i\gamma_1\varepsilon_k}\sum_{j\in E_i^{(k)}}|j\rangle\right)|i\rangle\\
&= \frac{1}{\sqrt{N}}\sum_{i=0}^{Q-1}\left(\sqrt{|G_i|}\,|g_i\rangle + \sum_k\sqrt{|E_i^{(k)}|}e^{-i\gamma_1\varepsilon_k}\,|e_i^{(j)}\rangle\right)|i\rangle\ ,
\end{aligned}
\tag{A.8}
$$

où la relation $\sum_{j\in G}|j\rangle\langle j|\sum_{j'=0}^{N}|j'\rangle = \sum_{j\in G}|j\rangle$ a été utilisé. Appliquons désormais U_D en calculant $|\psi_2\rangle = U_D\,|\psi_1\rangle$:

$$
\begin{aligned}
|\psi_2\rangle &= \left(\mathbb{1} - \frac{1}{N}(1-e^{-i\beta_1})\sum_{i',j'=0}^{N}|i'\rangle\langle j'|\right)\left(\frac{1}{\sqrt{N}}\sum_{i=0}^{Q-1}\left(\sqrt{|G_i|}\,|g_i\rangle + \sum_k\sqrt{|E_i^{(k)}|}e^{-i\gamma_1\varepsilon_k}\,|e_i^{(k)}\rangle\right)|i\rangle\right)\\
&= \frac{1}{\sqrt{N}}\left(\sum_{i=0}^{Q-1}\left(\sqrt{|G_i|}\,|g_i\rangle + \sum_k\sqrt{|E_i^{(k)}|}e^{-i\gamma_1\varepsilon_k}\,|e_i^{(k)}\rangle\right)|i\rangle\right.\\
&\quad\left.-\frac{1}{N}(1-e^{-i\beta_1})\sum_{i=0}^{Q-1}\left(|G_i| + \sum_k|E_i^{(k)}|e^{-i\gamma_1\varepsilon_k}\right)\sum_{i'=0}^{N}|i'\rangle\right)\\
&= \frac{1}{\sqrt{N}}\left(\sum_{i=0}^{Q-1}\left(\sqrt{|G_i|}\,|g_i\rangle + \sum_k\sqrt{|E_i^{(k)}|}e^{-i\gamma_1\varepsilon_k}\,|e_i^{(k)}\rangle\right)|i\rangle\right.\\
&\quad\left.-\frac{1}{N}(1-e^{-i\beta_1})\left(|G| + \sum_k|E^{(k)}|e^{-i\gamma_1\varepsilon_k}\right)\left(\sum_{i=0}^{Q-1}\left(\sqrt{|G_i|}\,|g_i\rangle + \sum_k\sqrt{|E_i^{(k)}|}\,|e_i^{(k)}\rangle\right)|i\rangle\right)\right)\\
&= \frac{1}{\sqrt{N}}\sum_{i=0}^{Q-1}\left(\sqrt{|G_i|}F_1(\gamma_1,\beta_1)\,|g_i\rangle + \sum_k\sqrt{|E_i^{(k)}|}\bar{F}_1^{(k)}(\gamma_1,\beta_1)\,|e_i^{(k)}\rangle\right)|i\rangle\ ,
\end{aligned}
\tag{A.9}
$$

où $F_1(\gamma_1,\beta_1) = 1 - \frac{1}{N}(1-e^{-i\beta_1})\left(|G| + \sum_{k'}|E^{(k')}|e^{-i\gamma_1\varepsilon_{k'}}\right)$ et $\bar{F}_1^{(k)}(\gamma_1,\beta_1) = e^{-i\gamma_1\varepsilon_k} - \frac{1}{N}(1-e^{-i\beta_1})\left(|G| + \sum_{k'}|E^{(k')}|e^{-i\gamma_1\varepsilon_{k'}}\right)$. Le cas initial a donc été prouvé.

Cas 2 (p couches) : Supposons que $|\psi\rangle = U_D U_P |\psi_0\rangle$ donne lieu à la relation suivante :

$$|\psi\rangle = \frac{1}{\sqrt{N}}\sum_{i=0}^{Q-1}\left(\sqrt{|G_i|}F_p(\gamma_1,\ldots,\gamma_p,\beta_1,\ldots,\beta_p)\,|g_i\rangle + \sum_k\sqrt{|E_i^{(k)}|}\bar{F}_p^{(k)}(\gamma_1,\ldots,\gamma_p,\beta_1,\ldots,\beta_p)\,|e_i^{(k)}\rangle\right)| \tag{A.10}$$

où

$$\begin{aligned} F_p(\gamma_1,\ldots,\gamma_p,\beta_1,\ldots,\beta_p) &= F_{p-1} - \frac{1}{N}(1-e^{-i\beta_p})\left(|G|F_{p-1} + \sum_{k'}|E^{(k')}|\bar{F}_{p-1}^{(k')}e^{-i\gamma_p\varepsilon_{k'}}\right) \\ \bar{F}_p^{(k)}(\gamma_1,\ldots,\gamma_p,\beta_1,\ldots,\beta_p) &= e^{-i\gamma_p\varepsilon_k}\bar{F}_{p-1}^{(k)} - \frac{1}{N}(1-e^{-i\beta_p})\left(|G|F_{p-1} + \sum_{k'}|E^{(k')}|\bar{F}_{p-1}^{(k')}e^{-i\gamma_p\varepsilon_{k'}}\right) \end{aligned} \tag{A.11}$$

Cas 3 ($p+1$ couches) : En utilisant la supposition précédente, appliquons U_P en calculant $|\psi_1\rangle = U_P |\psi_0\rangle$:

$$
\begin{aligned}
|\psi_1\rangle &= \left(\sum_{j\in G} |j\rangle\langle j| + \sum_k e^{-i\gamma_{p+1}\varepsilon_k} \sum_{j\in E^{(k)}} |j\rangle\langle j| \right) \left(\frac{1}{\sqrt{N}} \sum_{i=0}^{Q-1} \left(\sqrt{|G_i|} F_p |g_i\rangle + \sum_k \sqrt{|E_i^{(k)}|} \bar{F}_p^{(k)} |e_i^{(k)}\rangle \right) |i\rangle \right) \\
&= \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{Q-1} \sum_{j\in G_i} |j\rangle\langle j| \otimes |i\rangle\langle i| + \sum_{i=0}^{Q-1} \sum_k e^{-i\gamma_{p+1}\varepsilon_k} \sum_{j\in E_i^{(k)}} |j\rangle\langle j| \otimes |i\rangle\langle i| \right) \\
&\left(\sum_{i=0}^{Q-1} \left(F_p \sum_{j\in G_i} |j\rangle + \sum_k \bar{F}_p^{(k)} \sum_{j\in E_i^{(k)}} |j\rangle \right) |i\rangle \right) \\
&= \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{Q-1} F_p \sum_{j\in G_i} |j\rangle \otimes |i\rangle + \sum_{i=0}^{Q-1} \sum_k \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k} \sum_{j\in E_i^{(k)}} |j\rangle \otimes |i\rangle \right) \\
&= \frac{1}{\sqrt{N}} \sum_{i=0}^{Q-1} \left(F_p \sum_{j\in G_i} |j\rangle + \sum_k \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k} \sum_{j\in E_i^{(k)}} |j\rangle \right) |i\rangle \\
&= \frac{1}{\sqrt{N}} \sum_{i=0}^{Q-1} \left(\sqrt{|G_i|} F_p |g_i\rangle + \sum_k \sqrt{|E_i^{(k)}|} \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k} |e_i^{(k)}\rangle \right) |i\rangle
\end{aligned}
\tag{A.12}
$$

Appliquons ensuite U_D en calculant $|\psi_2\rangle = U_D |\psi_1\rangle$:

$$
\begin{aligned}
|\psi_2\rangle &= \left(\mathbb{1} - \frac{1}{N}(1 - e^{-i\beta_{p+1}}) \sum_{i',j'=0}^{N} |i'\rangle\langle j'|\right) \left(\frac{1}{\sqrt{N}} \sum_{i=0}^{Q-1} \left(\sqrt{|G_i|} F_p |g_i\rangle + \sum_k \sqrt{|E_i^{(k)}|} \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k} |e_i^{(k)}\rangle\right)\right) \\
&= \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{Q-1} \left(\sqrt{|G_i|} F_p |g_i\rangle + \sum_k \sqrt{|E_i^{(k)}|} \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k} |e_i^{(k)}\rangle\right) |i\rangle \right. \\
&\left. - \frac{1}{N}(1 - e^{-i\beta_{p+1}}) \sum_{i=0}^{Q-1} \left(|G_i| F_p + \sum_k |E_i^{(k)}| \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k}\right) \sum_{i'=0}^{N} |i'\rangle\right) \\
&= \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{Q-1} \left(\sqrt{|G_i|} F_p |g_i\rangle + \sum_k \sqrt{|E_i^{(k)}|} \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k} |e_i^{(k)}\rangle\right) |i\rangle \right. \\
&\left. - \frac{1}{N}(1 - e^{-i\beta_{p+1}}) \left(|G| F_p + \sum_k |E^{(k)}| \bar{F}_p^{(k)} e^{-i\gamma_{p+1}\varepsilon_k}\right) \sum_{i'=0}^{N} |i'\rangle\right) \\
&= \frac{1}{\sqrt{N}} \sum_{i=0}^{Q-1} \left(\sqrt{|G_i|} F_{p+1}(\gamma_1, \ldots, \gamma_{p+1}, \beta_1, \ldots, \beta_{p+1}) |g_i\rangle \right. \\
&\left. + \sum_k \sqrt{|E_i^{(k)}|} \bar{F}_{p+1}^{(k)}(\gamma_1, \ldots, \gamma_{p+1}, \beta_1, \ldots, \beta_{p+1}) e^{-i\gamma_{p+1}\varepsilon_k} |e_i^{(k)}\rangle\right) |i\rangle
\end{aligned}
\tag{A.13}
$$

où

$$
\begin{aligned}
F_{p+1}(\gamma_1, \ldots, \gamma_{p+1}, \beta_1, \ldots, \beta_{p+1}) &= F_p - \frac{1}{N}(1 - e^{-i\beta_{p+1}}) \left(|G| F_p + \sum_{k'} |E^{(k')}| \bar{F}_p^{(k')} e^{-i\gamma_{p+1}\varepsilon_{k'}}\right) \\
\bar{F}_{p+1}^{(k)}(\gamma_1, \ldots, \gamma_{p+1}, \beta_1, \ldots, \beta_{p+1}) &= e^{-i\gamma_{p+1}\varepsilon_k} \bar{F}_p^{(k)} - \frac{1}{N}(1 - e^{-i\beta_{p+1}}) \left(|G| F_p + \sum_{k'} |E^{(k')}| \bar{F}_p^{(k')} e^{-i\gamma_{p+1}\varepsilon_{k'}}\right)
\end{aligned}
\tag{A.14}
$$

La formule récursive a donc été prouvée par induction. Pour résoudre le sous-problème, modifions le circuit original tel que décrit à la section 4.2. Pour chacun des derniers q qubits :

1. Remplacer la porte H initial par une porte X conditionnée par la valeur à laquelle le qubit est fixé.
2. Enlever l'hamiltonien de forçage du qubit fixé.

Une formule récursive peut être trouvée par induction pour le circuit modifié en suivant exactement le même processus qu'effectué ci-haut. L'état final suit la même formule récursive que la formule A.5 avec les transformations suivantes :

$$\begin{aligned} \sum_{i=0}^{Q-1} |i\rangle &\to |c\rangle \ , \\ N &\to N_Q \, , \\ G &\to G_c \, , \\ E^{(k)} &\to E_c^{(k)} \, , \end{aligned} \tag{A.15}$$

où $N_Q = 2^{n-q}$. L'état final conserve alors les mêmes propriétés que le circuit GM-QAOA initial. □

Annexe B

Simulation de circuits quantiques avec les réseaux de tenseurs

La simulation classique d'états quantiques est tout sauf évidente. Ces états, appartenant à l'espace d'Hilbert, sont décrit par des vecteurs d'état de taille exponentielle empêchant ainsi leur caractérisation même pour des systèmes de taille modeste. Cette difficulté, présente dans de nombreux domaines tel l'apprentissage-machine, est aussi connue sous le nom de la *malédiction de la dimensionalité*. Cependant, la représentation de certains états physiques intéressants contient parfois de l'information superflue ou une structure inhérente. Par exemple, un état quantique général de n qubits requiert en théorie un vecteur d'état à 2^n bits, mais un état quantique non-intriqué nécessite seulement $2n$ bits en raison de l'absence de corrélation. Cette idée a alors mené au développement des méthodes de réseaux de tenseurs pour l'étude de système quantique à plusieurs corps en matière condensé afin d'obtenir une représentation des états quantiques plus efficace.

Cette annexe décrit seulement en surface les méthodes de réseaux de tenseur. Pour comprendre les différentes méthodes plus en profondeur, de nombreux tutoriels ont été écrits [82-84]

B.1 Réseaux de tenseurs

B.2 État en produit de matrices et opérateur en produit de matrices

B.3 Simulation de circuits quantiques

Bibliographie

1. Shor, P. *Algorithms for Quantum Computation : Discrete Logarithms and Factoring* in *Proceedings 35th Annual Symposium on Foundations of Computer Science* Proceedings 35th Annual Symposium on Foundations of Computer Science (nov. 1994), 124-134. https://ieeexplore.ieee.org/document/365700 (2025).
2. Cerezo, M. *et al.* Variational Quantum Algorithms. *Nature Reviews Physics* **3,** 625-644. issn : 2522-5820. https://www.nature.com/articles/s42254-021-00348-9 (2025) (sept. 2021).
3. Jerrum, M. R., Valiant, L. G. & Vazirani, V. V. Random Generation of Combinatorial Structures from a Uniform Distribution. *Theoretical Computer Science* **43,** 169-188. issn : 0304-3975. https://www.sciencedirect.com/science/article/pii/030439758690174X (2022) (1er jan. 1986).
4. Bärtschi, A. & Eidenbenz, S. *Grover Mixers for QAOA : Shifting Complexity from Mixer Design to State Preparation* in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* 2020 IEEE International Conference on Quantum Computing and Engineering (QCE) (oct. 2020), 72-82.
5. Farhi, E., Goldstone, J. & Gutmann, S. *A Quantum Approximate Optimization Algorithm* arXiv : 1411.4028 [quant-ph]. http://arxiv.org/abs/1411.4028 (2022). Prépubl.
6. Roth, D. On the Hardness of Approximate Reasoning. *Artificial Intelligence* **82,** 273-302. issn : 0004-3702. https://www.sciencedirect.com/science/article/pii/0004370294000921 (2024) (1er avr. 1996).
7. Sang, T., Bearne, P. & Kautz, H. *Performing Bayesian Inference by Weighted Model Counting* in *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1* (AAAI Press, Pittsburgh, Pennsylvania, 9 juill. 2005), 475-481. isbn : 978-1-57735-236-5. https://dl.acm.org/doi/10.5555/1619332.1619409 (2024).

8. Abramson, B., Brown, J., Edwards, W., Murphy, A. & Winkler, R. L. Hailfinder : A Bayesian System for Forecasting Severe Weather. *International Journal of Forecasting. Probability Judgmental Forecasting* **12,** 57-71. ISSN : 0169-2070. https://www.sciencedirect.com/science/article/pii/0169207095006648 (2024) (1er mars 1996).
9. Valiant, L. G. The Complexity of Enumeration and Reliability Problems. *SIAM J. Comput.* **8,** 410-421. ISSN : 0097-5397. https://doi.org/10.1137/0208032 (2025) (1er août 1979).
10. Duenas-Osorio, L., Meel, K., Paredes, R. & Vardi, M. Counting-Based Reliability Estimation for Power-Transmission Grids. *Proceedings of the AAAI Conference on Artificial Intelligence* **31.** ISSN : 2374-3468. https://ojs.aaai.org/index.php/AAAI/article/view/11178 (2024) (fév. 2017).
11. Jerrum, M. & Sinclair, A. Polynomial-Time Approximation Algorithms for the Ising Model. *SIAM J. Comput.* **22,** 1087-1116. ISSN : 0097-5397. https://doi.org/10.1137/0222066 (oct. 1993).
12. Baluta, T., Shen, S., Shinde, S., Meel, K. S. & Saxena, P. *Quantitative Verification of Neural Networks and Its Security Applications* in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (Association for Computing Machinery, New York, NY, USA, 6 nov. 2019), 1249-1264. ISBN : 978-1-4503-6747-9. https://doi.org/10.1145/3319535.3354245 (2024).
13. Aaronson, S. & Arkhipov, A. *The Computational Complexity of Linear Optics* in *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, USA, 6 juin 2011), 333-342. ISBN : 978-1-4503-0691-1. https://dl.acm.org/doi/10.1145/1993636.1993682 (2025).
14. Bouland, A., Fefferman, B., Nirkhe, C. & Vazirani, U. On the Complexity and Verification of Quantum Random Circuit Sampling. *Nature Physics* **15,** 159-163. ISSN : 1745-2481. https://www.nature.com/articles/s41567-018-0318-2 (2025) (fév. 2019).
15. PRIMES Is in P | Annals of Mathematics. https://annals.math.princeton.edu/2004/160-2/p12 (2024).
16. Cobham, A. in *Logic, Methodology and Philosophy of Science* (éd. Bar-Hillel, Y.) 24-30 (North-Holland Pub. Co., 1965).
17. Edmonds, J. Paths, Trees, and Flowers. *Canadian Journal of Mathematics* **17,** 449-467. ISSN : 0008-414X, 1496-4279. https://www.cambridge.org/core/journals/canadian-journal-of-mathematics/article/paths-trees-and-flowers/08B492B72322C4130AE800C0610E0E21 (2025) (jan. 1965).
18. Sipser, M. *Introduction to the Theory of Computation* 482 p. ISBN : 978-1-133-18779-0. Google Books : P3f6CAAAQBAJ (Cengage Learning, 27 juin 2012).

19. Carlson, J. A., Jaffe, A., Wiles, A., Institute, C. M. & Society, A. M. *The Millennium Prize Problems* 192 p. isbn : 978-0-8218-3679-8. Google Books : 7wJIPJ80RdUC (American Mathematical Soc., 2006).
20. Impagliazzo, R. & Paturi, R. On the Complexity of *K*-SAT. *Journal of Computer and System Sciences* **62,** 367-375. issn : 0022-0000. https://www.sciencedirect.com/science/article/pii/S0022000000917276 (2024) (1er mars 2001).
21. Valiant, L. G. The Complexity of Computing the Permanent. *Theoretical Computer Science* **8,** 189-201. issn : 0304-3975. https://www.sciencedirect.com/science/article/pii/0304397579900446 (2025) (1er jan. 1979).
22. Toda, S. PP Is as Hard as the Polynomial-Time Hierarchy. *SIAM Journal on Computing* **20,** 865-877. issn : 0097-5397. https://epubs.siam.org/doi/abs/10.1137/0220053 (2024) (oct. 1991).
23. Cook, S. A. *The Complexity of Theorem-Proving Procedures* in *Proceedings of the Third Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, USA, 3 mai 1971), 151-158. isbn : 978-1-4503-7464-4. https://dl.acm.org/doi/10.1145/800157.805047 (2024).
24. Levin, L. A. Universal Sequential Search Problems.
25. Marques-Silva, J. *Practical Applications of Boolean Satisfiability* in *2008 9th International Workshop on Discrete Event Systems* 2008 9th International Workshop on Discrete Event Systems (mai 2008), 74-80. https://ieeexplore.ieee.org/document/4605925 (2025).
26. Krom, M. R. The Decision Problem for a Class of First-Order Formulas in Which All Disjunctions Are Binary. *Mathematical Logic Quarterly* **13,** 15-20. issn : 1521-3870. https://onlinelibrary.wiley.com/doi/abs/10.1002/malq.19670130104 (2025) (1967).
27. Karp, R. M. in *Complexity of Computer Computations : Proceedings of a Symposium on the Complexity of Computer Computations, Held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and Sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department* (éd. Miller, R. E., Thatcher, J. W. & Bohlinger, J. D.) 85-103 (Springer US, Boston, MA, 1972). isbn : 978-1-4684-2001-2. https://doi.org/10.1007/978-1-4684-2001-2_9 (2025).
28. Schaefer, T. J. *The Complexity of Satisfiability Problems* in *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, USA, 1er mai 1978), 216-226. isbn : 978-1-4503-7437-8. https://dl.acm.org/doi/10.1145/800133.804350 (2025).

29. Froleyks, N., Heule, M., Iser, M., Järvisalo, M. & Suda, M. SAT Competition 2020. *Artificial Intelligence* **301,** 103572. ISSN : 0004-3702. https://www.sciencedirect.com/science/article/pii/S0004370221001235 (2025) (1er déc. 2021).

30. Vazirani, V. V. *Approximation Algorithms* ISBN : 978-3-642-08469-0 978-3-662-04565-7. http://link.springer.com/10.1007/978-3-662-04565-7 (2025) (Springer, Berlin, Heidelberg, 2003).

31. Lund, C. & Yannakakis, M. On the Hardness of Approximating Minimization Problems. *J. ACM* **41,** 960-981. ISSN : 0004-5411. https://dl.acm.org/doi/10.1145/185675.306789 (2025) (1er sept. 1994).

32. Biere, A., Biere, A., Heule, M., van Maaren, H. & Walsh, T. *Handbook of Satisfiability : Volume 185 Frontiers in Artificial Intelligence and Applications* 980 p. ISBN : 978-1-58603-929-5 (IOS Press, NLD, jan. 2009).

33. Davis, M., Logemann, G. & Loveland, D. A Machine Program for Theorem-Proving. *Commun. ACM* **5,** 394-397. ISSN : 0001-0782. https://dl.acm.org/doi/10.1145/368273.368557 (2025) (1er juill. 1962).

34. Kourtis, S., Chamon, C., Mucciolo, E. & Ruckenstein, A. E. Fast Counting with Tensor Networks. *SciPost Physics* **7,** 060. ISSN : 2542-4653. https://scipost.org/SciPostPhys.7.5.060 (2025) (12 nov. 2019).

35. Dudek, J. M., Dueñas-Osorio, L. & Vardi, M. Y. *Efficient Contraction of Large Tensor Networks for Weighted Model Counting through Graph Decompositions* arXiv : 1908.04381 [cs]. http://arxiv.org/abs/1908.04381 (2025). Prépubl.

36. Dudek, J. M. & Vardi, M. Y. *Parallel Weighted Model Counting with Tensor Networks* arXiv : 2006.15512 [cs]. http://arxiv.org/abs/2006.15512 (2025). Prépubl.

37. Stockmeyer, L. *The Complexity of Approximate Counting* in *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, USA, 1er déc. 1983), 118-126. ISBN : 978-0-89791-099-6. https://dl.acm.org/doi/10.1145/800061.808740 (2025).

38. Timme, M., van Bussel, F., Fliegner, D. & Stolzenberg, S. Counting Complex Disordered States by Efficient Pattern Matching : Chromatic Polynomials and Potts Partition Functions. *New Journal of Physics* **11,** 023001. ISSN : 1367-2630. https://dx.doi.org/10.1088/1367-2630/11/2/023001 (2025) (fév. 2009).

39. Brassard, G., HØyer, P. & Tapp, A. *Quantum Counting* in *Automata, Languages and Programming* (éd. Larsen, K. G., Skyum, S. & Winskel, G.) (Springer, Berlin, Heidelberg, 1998), 820-831. ISBN : 978-3-540-68681-1.

40. Achlioptas, D., Chtcherba, A., Istrate, G. & Moore, C. The Phase Transition in 1-in-k SAT and NAE 3-SAT. *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms* (4 mai 2001).

41. Raymond, J., Sportiello, A. & Zdeborová, L. The Phase Diagram of 1-in-3 Satisfiability Problem. *Physical Review E* **76,** 011101. issn : 1539-3755, 1550-2376. arXiv : cond-mat/0702610. http://arxiv.org/abs/cond-mat/0702610 (2023) (2 juill. 2007).
42. Zdeborová, L. *Statistical Physics of Hard Optimization Problems* arXiv : 0806.4112 [cond-mat]. http://arxiv.org/abs/0806.4112 (2023). Prépubl.
43. Trakhtenbrot, B. A. *Autoreducibility* in *Doklady Akademii Nauk* **192** (Russian Academy of Sciences, 1970), 1224-1227.
44. Selke, J. *Autoreducibility and Friends : About Measuring Redundancy in Sets* thèse de doct. (Master's thesis, Universität Hanover, 2006).
45. Hemaspaandra, L. A. in *Complexity and Approximation : In Memory of Ker-I Ko* (éd. Du, D.-Z. & Wang, J.) 19-47 (Springer International Publishing, Cham, 2020). isbn : 978-3-030-41672-0. https://doi.org/10.1007/978-3-030-41672-0_3.
46. Schnorr, C. *Optimal Algorithms for Self-Reducible Problems* in. International Colloquium on Automata, Languages and Programming (1976). https://www.semanticscholar.org/paper/Optimal-Algorithms-for-Self-Reducible-Problems-Schnorr/d640cec71aef6e8546979865492613716163 37b3 (2024).
47. Goldreich, O. Computational Complexity : A Conceptual Perspective. *SIGACT News* **39,** 35-39. issn : 0163-5700. https://dl.acm.org/doi/10.1145/1412700.1412710 (2025) (1er sept. 2008).
48. Khuller, S. & Vazirani, V. V. Planar Graph Coloring Is Not Self-Reducible, Assuming P $\neq$ NP. *Theoretical Computer Science* **88,** 183-189. issn : 0304-3975. https://www.sciencedirect.com/science/article/pii/030439759190081C (2025) (30 sept. 1991).
49. Jerrum, M. & Sinclair, A. Fast Uniform Generation of Regular Graphs. *Theoretical Computer Science* **73,** 91-100. issn : 0304-3975. https://www.sciencedirect.com/science/article/pii/030439759090164D (2025) (8 juin 1990).
50. Chakraborty, S., Meel, K. S. & Vardi, M. Y. *A Scalable and Nearly Uniform Generator of SAT Witnesses* arXiv : 1304.1584 [cs]. http://arxiv.org/abs/1304.1584 (2025). Prépubl.
51. Jerrum, M. *Counting, Sampling and Integrating : Algorithm and Complexity* isbn : 978-3-7643-6946-0 978-3-0348-8005-3. http://link.springer.com/10.1007/978-3-0348-8005-3 (2023) (Birkhäuser, Basel, 2003).
52. Born, M. & Fock, V. Beweis des Adiabatensatzes. *Zeitschrift für Physik* **51,** 165-180. issn : 0044-3328. https://doi.org/10.1007/BF01343193 (2024) (1er mars 1928).
53. Albash, T. & Lidar, D. A. Adiabatic Quantum Computing. *Reviews of Modern Physics* **90,** 015002. issn : 0034-6861, 1539-0756. arXiv : 1611.04471 [quant-ph]. http://arxiv.org/abs/1611.04471 (2024) (29 jan. 2018).

54. Messiah, A. *Quantum Mechanics* 1174 p. isbn : 978-0-486-40924-5. http://archive.org/details/quantummechanics0000mess (2024) (Mineola, N.Y. : Dover Publications, 1999).

55. Amin, M. H. S. Consistency of the Adiabatic Theorem. *Physical Review Letters* **102,** 220401. https://link.aps.org/doi/10.1103/PhysRevLett.102.220401 (2024) (3 juin 2009).

56. Farhi, E., Goldstone, J., Gutmann, S. & Sipser, M. *Quantum Computation by Adiabatic Evolution* arXiv : quant-ph/0001106. http://arxiv.org/abs/quant-ph/0001106 (2024). Prépubl.

57. Altshuler, B., Krovi, H. & Roland, J. Anderson Localization Makes Adiabatic Quantum Optimization Fail. *Proceedings of the National Academy of Sciences* **107,** 12446-12450. https://www.pnas.org/doi/10.1073/pnas.1002116107 (2025) (13 juill. 2010).

58. Nishimori, H. & Takada, K. Exponential Enhancement of the Efficiency of Quantum Annealing by Non-Stoquastic Hamiltonians. *Frontiers in ICT* **4.** issn : 2297-198X. https://www.frontiersin.org/journals/ict/articles/10.3389/fict.2017.00002/full (2025) (17 fév. 2017).

59. Hormozi, L., Brown, E. W., Carleo, G. & Troyer, M. Nonstoquastic Hamiltonians and Quantum Annealing of an Ising Spin Glass. *Physical Review B* **95,** 184416. https://link.aps.org/doi/10.1103/PhysRevB.95.184416 (2025) (15 mai 2017).

60. Wurtz, J. & Love, P. J. Counterdiabaticity and the Quantum Approximate Optimization Algorithm. *Quantum* **6,** 635. https://quantum-journal.org/papers/q-2022-01-27-635/ (2025) (27 jan. 2022).

61. Farhi, E., Gamarnik, D. & Gutmann, S. *The Quantum Approximate Optimization Algorithm Needs to See the Whole Graph : A Typical Case* arXiv : 2004.09002 [quant-ph]. http://arxiv.org/abs/2004.09002 (2025). Prépubl.

62. Egger, D. J., Mareček, J. & Woerner, S. Warm-Starting Quantum Optimization. *Quantum* **5,** 479. https://quantum-journal.org/papers/q-2021-06-17-479/ (2025) (17 juin 2021).

63. Lucas, A. Ising Formulations of Many NP Problems. *Frontiers in Physics* **2.** issn : 2296-424X. https://www.frontiersin.org/article/10.3389/fphy.2014.00005 (2014).

64. Lodewijks, B. *Mapping NP-hard and NP-complete Optimisation Problems to Quadratic Unconstrained Binary Optimisation Problems* arXiv : 1911.08043 [cs]. http://arxiv.org/abs/1911.08043 (2025). Prépubl.

65. Hadfield, S. *et al.* From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz. *Algorithms* **12,** 34. issn : 1999-4893. https://www.mdpi.com/1999-4893/12/2/34 (2022) (2 fév. 2019).

66. McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R. & Neven, H. Barren Plateaus in Quantum Neural Network Training Landscapes. *Nature Communications* **9,** 4812. ISSN : 2041-1723. https://www.nature.com/articles/s41467-018-07090-4 (2025) (16 nov. 2018).

67. Larocca, M. *et al. A Review of Barren Plateaus in Variational Quantum Computing* arXiv : 2405.00781 [quant-ph]. http://arxiv.org/abs/2405.00781 (2025). Prépubl.

68. Anschuetz, E. R. & Kiani, B. T. Quantum Variational Algorithms Are Swamped with Traps. *Nature Communications* **13,** 7760. ISSN : 2041-1723. https://www.nature.com/articles/s41467-022-35364-5 (2025) (15 déc. 2022).

69. Bittel, L. & Kliesch, M. Training Variational Quantum Algorithms Is NP-Hard. *Physical Review Letters* **127,** 120502. https://link.aps.org/doi/10.1103/PhysRevLett.127.120502 (2022) (17 sept. 2021).

70. Sack, S. H. & Serbyn, M. Quantum Annealing Initialization of the Quantum Approximate Optimization Algorithm. *Quantum* **5,** 491. https://quantum-journal.org/papers/q-2021-07-01-491/ (2022) (1er juill. 2021).

71. Matsuda, Y., Nishimori, H. & Katzgraber, H. G. Ground-State Statistics from Annealing Algorithms : Quantum versus Classical Approaches. *New Journal of Physics* **11,** 073021. ISSN : 1367-2630. https://doi.org/10.1088/1367-2630/11/7/073021 (2022) (juill. 2009).

72. Mandrà, S., Zhu, Z. & Katzgraber, H. G. Exponentially Biased Ground-State Sampling of Quantum Annealing Machines with Transverse-Field Driving Hamiltonians. *Phys. Rev. Lett.* **118,** 070502. https://link.aps.org/doi/10.1103/PhysRevLett.118.070502 (2022) (fév. 2017).

73. Sundar, B., Paredes, R., Damanik, D. T., Dueñas-Osorio, L. & Hazzard, K. R. A. *A Quantum Algorithm to Count Weighted Ground States of Classical Spin Hamiltonians* arXiv : 1908.01745 [quant-ph]. http://arxiv.org/abs/1908.01745 (2025). Prépubl.

74. Xie, N. *et al.* Performance Upper Bound of a Grover-mixer Quantum Alternating Operator Ansatz. *Physical Review A* **111,** 012401. https://link.aps.org/doi/10.1103/PhysRevA.111.012401 (2025) (2 jan. 2025).

75. Viger, F. & Latapy, M. *Efficient and Simple Generation of Random Simple Connected Graphs with Prescribed Degree Sequence* in *Computing and Combinatorics* (éd. Wang, L.) (Springer, Berlin, Heidelberg, 2005), 440-449. ISBN : 978-3-540-31806-4.

76. Gray, J. Quimb : A Python Package for Quantum Information and Many-Body Calculations. *Journal of Open Source Software* **3,** 819. ISSN : 2475-9066. https://joss.theoj.org/papers/10.21105/joss.00819 (2023) (4 sept. 2018).

77. Virtanen, P. *et al.* SciPy 1.0 : Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **17,** 261-272. issn : 1548-7105. https://www.nature.com/articles/s41592-019-0686-2 (2025) (mars 2020).

78. Sharma, S., Roy, S., Soos, M. & Meel, K. S. GANAK : A Scalable Probabilistic Exact Model Counter. *Electronic proceedings of IJCAI 2019,* 1169-1176. https://www.ijcai.org/Proceedings/2019/163 (2023) (2019).

79. Eén, N. & Sörensson, N. *An Extensible SAT-solver* in *Theory and Applications of Satisfiability Testing* (éd. Giunchiglia, E. & Tacchella, A.) (Springer, Berlin, Heidelberg, 2004), 502-518. isbn : 978-3-540-24605-3.

80. Audemard, G. & Simon, L. *Predicting Learnt Clauses Quality in Modern SAT Solvers* in *Proceedings of the 21st International Joint Conference on Artificial Intelligence* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 11 juill. 2009), 399-404. https://dl.acm.org/doi/10.5555/1661445.1661509 (2023).

81. Ignatiev, A., Morgado, A. & Marques-Silva, J. *PySAT : A Python Toolkit for Prototyping with SAT Oracles* in *Theory and Applications of Satisfiability Testing – SAT 2018* (éd. Beyersdorff, O. & Wintersteiger, C. M.) (Springer International Publishing, Cham, 2018), 428-437. isbn : 978-3-319-94144-8.

82. Bridgeman, J. C. & Chubb, C. T. Hand-Waving and Interpretive Dance : An Introductory Course on Tensor Networks. *Journal of Physics A : Mathematical and Theoretical* **50,** 223001. issn : 1751-8113, 1751-8121. arXiv : 1603.03039 [quant-ph]. http://arxiv.org/abs/1603.03039 (2025) (2 juin 2017).

83. Biamonte, J. & Bergholm, V. *Tensor Networks in a Nutshell* arXiv : 1708.00006 [quant-ph]. http://arxiv.org/abs/1708.00006 (2025). Prépubl.

84. Baker, T. E., Desrosiers, S., Tremblay, M. & Thompson, M. P. Méthodes de Calcul Avec Réseaux de Tenseurs En Physique (Basic Tensor Network Computations in Physics). *Canadian Journal of Physics* **99,** 207-221. issn : 0008-4204, 1208-6045. arXiv : 1911.11566 [quant-ph]. http://arxiv.org/abs/1911.11566 (2025) (avr. 2021).