CentraleSupélec

université PARIS-SACLAY

Autour des Diagrammes de Décision Quantiques

Rapport – Parcours recherche – Projet S6

Élève : Malo Leroy
Encadrant : Renaud Vilmart

Table des matières

11 juin 2024

Chapitre 1

Introduction

1.1 Contexte et objectif

L'**informatique quantique** est un domaine en plein essor. Cette technologie permettant de manipuler des qubits à la place des bits qui sont à la base de l'informatique actuelle, ouvrent la voie à des algorithmes plus performants que les algorithmes classiques. Les machines quantiques exécutant ces algorithmes étant encore en développement et d'un coût important, il existe un besoin d'outils de simulation et de vérification d'algorithmes quantiques par des machines classiques. L'objectif de ce projet est de proposer un modèle sur les diagrammes de décision quantiques, en s'appuyant sur les travaux existants, et de les simuler pour en étudier les performances.

1.2 État de l'art

Informatique quantique

Le premier postulat de la physique quantique, le **principe de superposition**, énonce que l'espace des états d'un système quantique est un espace de Hilbert. Par conséquent, si un *bit* ne peut classiquement être que dans un état $|0\rangle$ ou $|1\rangle$, son homologue quantique, le **qubit**, peut être dans une superposition de ces deux états.

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

où α, β sont des coefficients complexes. Le second postulat, le *principe de mesure*, stipule que lorsqu'un qubit est mesuré, il est projeté sur un des états de base $|0\rangle$ ou $|1\rangle$ avec une probabilité $|\alpha|^2$ ou $|\beta|^2$ respectivement.

Les états à n qubits peuvent être représentés par des **vecteurs** de dimension 2^n, les états de l'ensemble constitué de deux systèmes étant ceux obtenus par produit tensoriel (*produit de Kronecker*) d'un état du premier système et d'un état du second. C'est ce nombre exponentiel de paramètres complexes pour un état à plusieurs qubits qui rend l'étude classique d'algorithmes quantiques difficile, puisque les états prennent une taille exponentiellement grande en mémoire.

Diagrammes de décision

La structure de **diagramme de décision** est une structure de données développée à partir des années 1970. Elle est depuis devenue largement utilisée en informatique, notamment pour rendre plus compacte la représentation de **fonctions binaires** [5].

Prenons par exemple la fonction $f(x_1, x_2, x_3) = x_1 \vee (x_2 \wedge x_3)$. La représentation d'une telle fonction se fait dans le cas général à l'aide d'une *table de vérité*. On peut utiliser une représentation plus compacte en utilisant un diagramme de décision, comme le montre la Figure 1.1, où les fils gauches sont signalés par des flèches pointillées et les fils droits sont signalés par des flèches continues.

Les diagrammes de décision prennent avantage de la **structure** interne de la donnée (ici, une fonction booléenne). On note d'une part que les labels ne sont pas nécessaires pour reconstituer les valeurs prises par la fonction. D'autre part, dans le pire des cas, c'est-à-dire celui où la fonction ne possède aucune structure permettant une réduction, la taille du diagramme de décision (son nombre de branches) vaut $2^{n+1} - 2$ ce qui comme le nombre de valeurs 2^n à stocker dans une table de vérité, est **exponentiel** en n. Dans le pire cas, les diagrammes de décision ne permettent pas d'amélioration, mais ne sont pas non plus asymptotiquement pire que les tables de vérité.

x_1	x_2	x_3	$f(x_1, x_2)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

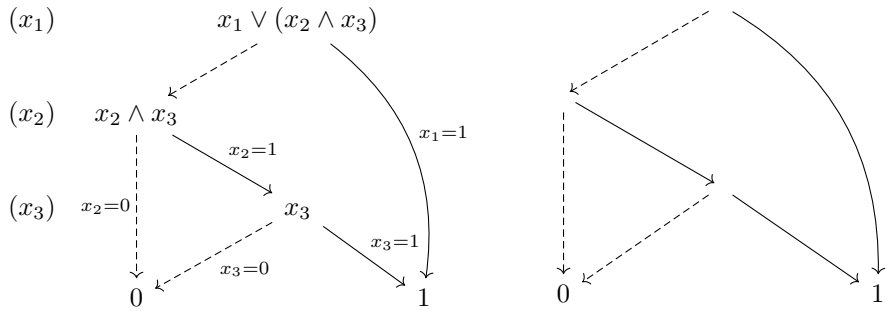


FIGURE 1.1 – (a) Table de vérité (b) Diagramme de décision (c) Diagramme de décision sans labels

Interprétation abstraite

L'interprétation abstraite est une méthode générale consistant à traiter des propriétés de programmes informatiques en les abstrayant. Elle a été introduite par Patrick et Radhia Cousot en 1977 [3]. Elle peut aussi être utilisée pour résoudre des problèmes ou calculer plus rapidement.

Étudions le problème suivant : on cherche à déterminer le signe de l'expression -12 * 7 - 13. On pourrait calculer le résultat de cette expression, mais on peut aussi remarquer que -12 et -13 sont négatifs et que 7 est positif, d'où -12 * 7 négatif, ainsi l'expression est négative. Plus formellement, on est parti des **éléments concrets** que sont -12, 7 et -13 et on les a remplacés par les **éléments abstraits** "positif" $\oplus$ et "négatif" $\ominus$, sur lesquels on définit les opérations de somme et de produit selon les règles bien connues.

Il existe de nombreux cas d'utilisation de l'interprétation abstraite [7], pour la compilation de programmes par exemple. Nous l'utiliserons dans ce projet pour simplifier les diagrammes de décision quantiques quitte à perdre une partie de l'information qu'ils contiennent. C'est aussi le cas pour l'exemple précédent : abstraire les éléments par leur signe ne permet pas toujours de déterminer le signe de l'expression.

1.3 Contribution

Les différents concepts présentés dans l'état de l'art permettent tous d'améliorer la vitesse de calcul ou la taille en mémoire d'une donnée. Il s'agit donc de **combiner ces concepts** dans ce projet, en ajoutant une dimension supplémentaire, l'additivité : le fait pour un diagramme d'avoir plusieurs fils droits (respectivement gauches), l'interprétation étant que le « fils droit effectif » est la somme des fils droits (respectivement gauches) du diagramme. L'objectif du projet était donc de proposer un modèle de diagrammes de décision quantiques abstraits et additifs, et de les simuler pour en étudier les performances en comparaison d'autres modèle.

De manière préliminaire, une étude des intervalles complexes polaires a été réalisée, ce qui n'avait pas été fait jusqu'à présent. Un **modèle** mathématique de la structure de données et de méthodes de réduction ont été formalisés, et des algorithmes de réduction ont été développés pour simplifier ces diagrammes. Une **implémentation** de ces algorithmes a été réalisée, sans se baser sur des bibliothèques existantes (mise à part pour les tests unitaires).

1.4 Structure du rapport

Le Chapitre 2 présente le modèle théorique de diagrammes de décision quantiques qui a été développé, y compris les algorithmes de réduction qui accompagnent celui-ci. Ensuite, le Chapitre 3 discute de l'implémentation réalisée. Finalement, une courte conclusion est présentée dans le Chapitre 4.

Un document plus complet que ce rapport destiné à être publié et rédigé en anglais, est disponible en ligne. [4] D'éventuels points théoriques succinctement présentés dans ce rapport et ou théorèmes dont la preuve n'est pas spécifiée sont détaillés dans ce document.

Chapitre 2

Modèle théorique

Le modèle théorique proposé est celui des **diagrammes de décision quantiques additifs abstraits**.

2.1 Arithmétique des intervalles

L'arithmétique des intervalles dans le cadre de l'interprétation abstraite est proche de l'exemple de calcul sur les signes de la section 1.2. L'arithmétique des intervalles est utilisée pour déterminer un ensemble de valeurs possibles pour une expression mathématique en utilisant des intervalles pour les valeurs des variables.

L'arithmétique des intervalles réels a été étudiée [8], et l'arithmétique des intervalles complexes cartésien a fait l'objet d'études légères par le passé [6]. Au cours de ce projet, un travail a été réalisé pour explorer les possibilités de l'arithmétique des **intervalles complexes cartésiens et polaires**.

Les intervalles complexes cartésiens sont définis par un intervalle pour la partie réelle et un intervalle pour la partie imaginaire. Les intervalles complexes polaires sont définis par un intervalle pour le module et un intervalle pour l'argument. Dans un cas comme dans l'autre, l'équivalent abstrait d'une **opération** $*$ est définie sur des éléments α et β de l'ensemble des intervalles cartésiens $\mathcal{A}_0$ ou polaires $\mathcal{S}_0$ par

$$\alpha * \beta = \bigcap_{\gamma \supset \alpha \circledast \beta \text{ et } \gamma \in \mathcal{A}_0} \gamma \quad \text{où} \quad \alpha \circledast \beta = \{a * b; a \in \alpha, b \in \beta\}$$

Les opérations de somme, de produit et d'union ainsi construites sont sur-approximées : elles garantissent que l'ensemble des valeurs possibles est inclus dans l'ensemble abstrait, et d'avoir un intervalle pour résultat. Ces opérations ont des propriétés, de distributivité par exemple, parfois très différentes des nombres complexes qu'ils représentent. Des propriétés sur les intervalles cartésiens et polaires sont énoncées et démontrées dans le document annexe [4].

2.2 États abstraits et diagrammes

Les **états** abstraits à n qubits sont définis comme des 2^n-uplets d'intervalles complexes, on note leur ensemble $\mathcal{A}_n$ ou $\mathcal{S}_n$. On définit sur les états la relation d'ordre d'inclusion des états par l'inclusion des produits cartésiens des intervalles les composant.

Les **diagrammes** sont définis de manière récursive. Le seul diagramme de hauteur 0 est $\boxed{1}$, puis si l'ensemble $\mathcal{D}_n$ des diagrammes de hauteur n est défini, les diagrammes de hauteur $n+1$ peuvent avoir un nombre fini de fils gauches dans $\mathcal{D}_n$ et un nombre fini de fils droits dans $\mathcal{D}_n$, chacun étant associé à une amplitude abstraite sur la branche dans $\mathcal{A}_0$

$$\mathcal{D}_{n+1} = \mathscr{P}_f(\mathcal{A}_0 \times \mathcal{D}_n) \times \mathscr{P}_f(\mathcal{A}_0 \times \mathcal{D}_n)$$

On peut ainsi définir la fonction d'évaluation $\mathcal{E} : \mathcal{D}_n \to \mathcal{A}_n$ sur les diagrammes (une définition similaire est possible en utilisant les intervalles polaires), ce qui permet de définir une relation d'ordre $\leq$ sur les diagrammes par inclusion des ensembles abstraits qu'ils représentent.

2.3 Approximations

L'un des objectifs pour cette structure de données est de transformer un diagramme en un autre incluant le diagramme initial et de taille plus faible. Sur les diagrammes de décision abstraits ont été développés deux algorithmes, à partir d'une relation de fusion.

Puisque traiter les diagrammes de manière globale est difficile, on réalise les approximations de manière locale. Plus formellement, une **approximation globale** est une fonction $g : \mathcal{D}_n \to \mathcal{D}_n$ telle que

$$\forall D \in \mathcal{D}_n, D \leq g(D)$$

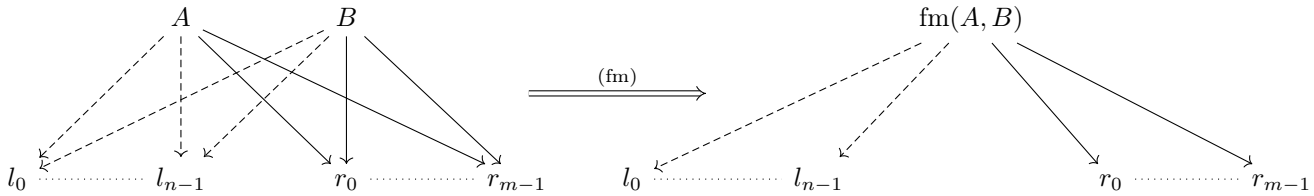
En pratique, on a surtout développé une **approximation par fusion**, qui est une fonction f telle que

$$\begin{cases} \forall A \neq B \in \mathcal{D}_n, A \leq f(A,B) \text{ and } B \leq f(A,B) \\ \forall A \in \mathcal{D}_n, f(A,A) = A \end{cases}$$

Le **théorème de fusion** indique que si l'on dispose d'une approximation par fusion, alors on dispose d'une approximation globale. Il simplifie grandement les preuves ultérieures, puisque l'on peut se contenter de démontrer la propriété d'approximation par fusion pour montrer l'approximation globale.

2.4 Réduction

Deux algorithmes de réduction ont été développés. Ils se reposent tous deux sur l'approximation par fusion fm (pour **force merge**) suivante



où si ampl(A, x) est l'amplitude abstraite sur le lien entre A et x, alors les nouvelles amplitudes abstraites sont définies par la formule suivante

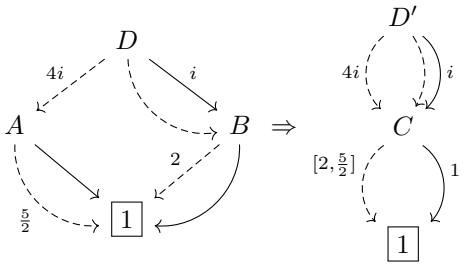
$$\forall x \in \{l_0, ..., l_{k-1}, r_0, ..., r_{m-1}\}, \text{ampl}(\text{fm}(A,B), x) = \text{ampl}(A,x) \sqcup \text{ampl}(B,x)$$

où $\sqcup$ est l'opération d'union des intervalles complexes (cartésiens ou polaires). Cette approximation par fusion fonctionne en pratique y compris sur des diagrammes n'ayant a priori pas de descendance commune, puisqu'il suffit d'ajouter des branches avec un poids nul pour se ramener à ce cas.

2.5 Exemple

Considérons le diagramme suivant, où tous les fils (gauche ou droit) sont $\boxed{1}$ et où les branches sans amplitude écrite sont d'amplitude 1. Appliquer fm sur A et B permet fait passer le diagramme additif initial à un diagramme additif abstrait $D' \geq D$.

Ici, la fusion permettrait ensuite de fusionner les deux branches gauches de D' pour obtenir un diagramme non additif.



Chapitre 3

Implémentation

L'implémentation a été réalisée en **langage C++**. Ce choix a été motivé par plusieurs raisons : la performance, la maturité du langage et son utilisation dans plusieurs projets historiques du domaine [1], et la proximité avec le langage C, déjà bien connu.

3.1 Structure du code

L'implémentation utilise largement la **programmation orientée objet**, en définissant des classes pour les objets manipulés. Les classes principales sont les suivantes, du plus bas au plus haut niveau :

— Intervalles réels (quelconques, de réels positifs ou modulo 2π), et complexes (cartésiens ou polaires)
— **Diagrammes** (par défaut, additifs et abstraits)
— Les branches, qui contiennent un lien (**pointeur**) vers un digramme de destination et un intervalle complexe, et dont la définition typée par la hauteur est mutuellement récursive avec celle des diagrammes

Les diagrammes et les branches sont définis avec des **templates** par rapport à leur hauteur. Ce point a fait l'objet d'hésitations, et a été retenu pour plusieurs raisons :

— Le typage des diagrammes et des fonctions d'interprétation, de sélection ou de réduction est plus fort : un diagramme de hauteur 3 ne peut avoir comme fils que des diagrammes de hauteur 2
— La terminaison du parcours récursif d'un diagramme est garantie : un diagramme de hauteur nulle ne pouvant avoir de fils, et le parcours se faisant souvent par réduction de la hauteur

Les classes sont organisées en **espaces de noms** pour éviter les conflits de noms, et définies dans des fichiers séparés. Des fonctions de **réduction** servent ensuite à réduire les diagrammes, en utilisant les fonctions de sélection.

3.2 Outils

Le code source est versionné à l'aide du logiciel *Git*, et est disponible sur la plateforme *GitHub* [4]. Ce projet fait l'objet d'une **intégration continue** utilisant *GitHub Actions*, automatisant la validation des tests réalisés en utilisant la bibliothèque open-source *Google Test*. Le projet est compilé à l'aide du compilateur *GCC*, ceci étant facilité par l'outil *CMake*.

Chapitre 4

Conclusion

4.1 Travail réalisé

Le modèle qui a été développé, avec algorithme de réduction, permet de limiter autant que souhaité la taille d'un état en mémoire, au prix d'une perte de précision. Le cadre mathématique de celui-ci a été défini, et des algorithmes de réduction ont étés prouvés sur cette structure de données.

L'implémentation, n'a pas encore fourni de résultats expérimentaux significatifs, mais permet déjà de simuler des diagrammes de décision quantiques de manière performante. Sa robustesse est assurée par des tests unitaires.

Le travail réalisé sur cette période est donc encourageant : malgré des difficultés rencontrées, tant d'un point de vue technique en programmation que dans la définition du modèle, ayant parfois amené à revenir en arrière avant de trouver une bonne définition de l'erreur par exemple, les résultats théoriques obtenus sont prometteurs.

4.2 Perspectives

Plusieurs perspectives peuvent être envisagées pour prolonger les travaux réalisés. D'une part, puisque plusieurs choix dans la définition du modèle ont étés faits de manière arbitraire parmi plusieurs options possibles (fonction d'erreur notamment), il est pertinent de réaliser de nombreuses simulations dans des configurations différentes afin de déterminer les choix les plus pertinents.

Nous pourrons aussi étendre le modèle avec d'autres concepts, comme les automates d'arbres pour la **vérification** [2] ou les diagrammes de décisions et applications localement inversibles [9]. Enfin, l'implémentation aurait à gagner d'être plus facile d'utilisation, par exemple avec une **interface graphique** et un **lecteur QASM** (équivalent quantique du format assembleur en informatique classique). Ces axes de travail pourront être explorés au premier semestre de la deuxième année du cursus CentraleSupélec si le projet est prolongé.

Bibliographie

[1] Benjamin Bichsel, Anouk Paradis, Maximilian Baader, and Martin Vechev. Abstraqt : Analysis of quantum circuits via abstract stabilizer simulation. *Quantum*, 7 :1185, November 2023.

[2] Yu-Fang Chen, Kai-Min Chung, Ondřej Lengál, Jyun-Ao Lin, Wei-Lun Tsai, and Di-De Yen. An automata-based framework for verification and bug hunting in quantum circuits (technical report), 2023.

[3] P. Cousot and R. Cousot. Abstract interpretation : a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977. ACM Press, New York, NY.

[4] Malo Leroy. Abstract additive quantum decision diagrams. `https://github.com/Firefnix/coto/`, June 2024.

[5] Shin-ichi Minato. *Binary decision diagrams and applications for VLSI CAD.* Kluwer international series in engineering and computer science. VLSI, computer architecture, and digital signal processing. Springer New York, NY, 1996.

[6] J. Rokne and P. Lancaster. Complex interval arithmetic. *Commun. ACM*, 14(2) :111–112, feb 1971.

[7] Mads Rosendahl. *Introduction to abstract interpretation.* Computer Science University of Copenhagen, 1995.

[8] Teruo Sunaga. Theory of an interval algebra and its application to numerical analysis [Reprint of Res. Assoc. Appl. Geom. Mem. 2 (1958), 29–46]. *Japan Journal of Industrial and Applied Mathematics*, 26(2-3) :125 – 143, 2009.

[9] Lieuwe Vinkhuijzen, Tim Coopmans, David Elkouss, Vedran Dunjko, and Alfons Laarman. Limdd : A decision diagram for simulation of quantum computing including stabilizer states. *Quantum*, 7 :1108, September 2023.