Qbricks, un environnement noyau pour la vérification de programmes quantiques

Christophe Chareton, Sébastien Bardin, François Bobot, Valentin Perelle
CEA, LIST, Université Paris-Saclay, France

Benoît Valiron
LRI, Centrale Supélec, Université Paris-Saclay, Orsay, France

Après des progrés importants [1] en algorithmique depuis 1982 [2] et l'avènement des premières machines élaborées par les grands investisseurs (Google, IBM, D-Wawe, etc), les différents stades du développement en informatique quantique sont désormais identifiés et commencent à être explorés : langages de programmation (Q#, liqui $|\rangle$, Quipper,etc), langages assembleurs (QASM), optimisation etc. Une question cruciale émerge de ces avancées : comment un développeur peut-il s'assurer de la correction des programmes qu'il écrit ?

La complexité (espaces vectoriels complexes de dimension exponentielle, intrication, etc) des structures de données quantiques rend en effet essentiel de disposer de méthodes efficaces de vérification. Il est pourtant impossible de transposer les stratégies de tests utilisées en informatique classique (mesure destructive, indéterminisme quantique, calcul quantique hors de portée d'une simulation classique, etc).

Nous pensons que la solution à ces impasses doit venir de la vérification formelle de programmes : ces techniques sont statiques, elles permettent de fournir des garanties absolues sur le comportement de systèmes et elles s'appliquent à des espaces d'états infinis.

Notre but est donc de développer une solution complète de programmation quantique formellement certifiée. Celle-ci doit offrir des possibilités d'écriture de programmes directement dérivés de la description d'algorithmes issue de la littérature. Elle doit permettre une distinction claire entre un programme et sa spécification, fournir des certifications insensibles à l'échelle pour des familles de circuits paramétrées et permettre, autant que faire se peut, un traitement automatique des preuves.

Dans ces buts nous développons l'environnement *Qbricks*. Il contient notamment un langage, embarqué dans l'environnement Why3 [3], d'écriture et de manipulation de (familles de) circuits quantiques. Ce langage est fourni avec des outils sémantiques pour la spécification des circuits. Il s'agit d'une implémentation complète, à la fois d'une sémantique fonctionnelle utilisée comme interface utilisateur et pour la preuve formelle, et de la sémantique matricielle, la référence standard du domaine, qui apporte un critère de correction sémantique. Nous fournissons une preuve formelle d'équivalence entre ces deux formalismes. *Qbricks* contient également un ensemble de mécanismes de simplification sémantique locale, qui permettent de faciliter les preuves de correction pour des circuits respectant certaines contraintes.

Grâce à *Qbricks*, nous présentons une implémentation paramétrée, vérifiée et invariante d'échelle du circuit d'estimation de phase. Il s'agit de la partie quantique de l'algorithme de Shor [4] pour la factorisation des nombres entiers. Ce circuit est aussi au coeur de nombreux algorithmes quantiques emblématiques (inversion de matrice, simulation quantique etc). Notre implémentation est la première preuve de correction invariante d'échelle pour une routine quantique non triviale.

Références

[1] Quantum algorithm zoo. `https://quantumalgorithmzoo.org/`.

[2] Richard P. Feynman. Simulating physics with computers. *j-INT-J-THEOR-PHYS*, 21, 1982.

[3] Jean-Christophe Filliâtre and Andrei Paskevich. Why3 - where programs meet provers. In *ESOP 2013*, pages 125–128, 2013.

[4] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5), 1997.