Практические языки программирования квантовых вычислений.

Поле квантовых алгоритмов динамично. Тем не менее, до недавнего времени отсутствовали языки программирования для описания квантовых вычислений на практическом уровне. В [1] решают эту проблему путем введения масштабируемого, функционального, квантового языка программирования высокого уровня Quipper, который был использован для программирования набора разнообразных нетривиальных квантовых алгоритмов. Он ориентирован на модели вычислений, которые используют классический компьютер для управления квантовым устройством, но не зависит от какой-либо конкретной модели квантового оборудования. Язык Quipper эффективен и прост. Он открывает возможность использования формальных методов для анализа квантовых алгоритмов. Наиболее ранние компьютеры было трудно запрограммировать. Трудность обусловлена отчасти необходимостью выражать алгоритмы в лексике подходящей для конкретного оборудования. Введение символических языков программирования позволяет специфицировать алгоритмы в форме более пригодной для понимания человеком, а затем переводить эту спецификацию в форму выполняемую машиной. Таким образом, языки программирования, предполагается, исполняют важную роль моста через семантический разрыв между человеком и вычислительным устройством. Это было достигнуто, в частности, посредством абстракций высокого уровня и автоматизированных вычислений. Квантовые вычисления, которые появились в конце 20-го века, это вычислительная парадигма основанная на законах квантовой физики. В литературе было наглядно продемонстрировано, что квантовые вычисления могут, теоретически, превзойти классические вычисления для некоторых классов вычислительных задач. Конструирование новых квантовых алгоритмов является динамичной областью, о чем свидетельствует "зоопарк" квантовых алгоритмов [2], который содержит ссылки на 45 алгоритмов и 160 статей, из которых не менее чем 14 были написаны в 2011 и 2012 годах. Хотя квантовые вычисления еще не готовы к переходу от теории к практике, тем не менее, можно обоснованно догадываться какую форму, возможно, квантовый компьютер примет, или, что более важно для дизайна языка программирования, по какому интерфейсу можно будет взаимодействовать с таким квантовым компьютером. Естественно применить уроки, извлеченные из программирования классических вычислений к квантовым вычислениям. Эта статья является ступенькой в достижении этой задачи. Подойдем к квантовым вычислениям с точки зрения программиста, решая как надо проектировать язык программирования, который может реализовать реальные квантовые алгоритмы эффективно, четко и с поддержкой? Введем Quipper, декларативный язык с монадической операционной семантикой [15], являющийся лаконичным, выразительным и масштабируемым, с солидной теоретической основой. Когда говорят, что Quipper "масштабируем", имеют в виду, что он выходит далеко за рамки игрушечных алгоритмов и простых доказательств. Многие фактически квантовые алгоритмы в литературе на порядки сложнее, чем то, что может быть реально реализовано в ранее существовавших квантовых языках программирования. Испытания реализации Quipper ставили на семи нетривиальных квантовых алгоритмах известных из литературы:

- Бинарное спаянное дерево (Binary Welded Tree, BWT), arXiv:quant-ph/0209131. Поиск обозначенного узла в графе.
- Булева формула (Boolean Formula, BF), arXiv:0704.3628 и arXiv:quant-ph/0703015, любая формула AND-OR размера N может быть вычислена за время $\mathbf{n^{1/2+o(1)}}$ на квантовом компьютере. Реализована версия, которая вычисляет выигрышную стратегию в настольной игре гекс.
- Порядок класса (Class Number, CL), Proceedings of the 34th ACM Symposium on Theory of Computing, 2002. Квантовый алгоритм для вычисления уравнения Пелля и главного идеала за полиномиальное время.

- Оценка основного состояния квантово-механической системы (Ground State Estimation, GSE), Molecular Physics, 109(5):735–750, 2011. Вычисление энергетических уровней для конкретной молекулы.
- Квантовые линейные системы (Quantum Linear Systems, QLS), arXiv:0811.3171. Решение линейной системы уравнений.
- Кратчайший уникальный вектор (Unique Shortest Vector, USV), arXiv:cs/0304005. Поиск кратчайшего вектора среди имеющихся вариантов.
- Поиск треугольника (Triangle Finding, TF), arXiv:quant-ph/0310134. Поиск треугольников в насыщенном графе. [3]

Эти алгоритмы были выбраны IARPA, в контексте его программы [4], чтобы обеспечить достаточно представительную выборку существующих алгоритмов. Они делают использование широкого спектра квантовых примитивов, таких как амплитуды усиления [14], квантовые блуждания, квантовое преобразование Фурье и квантовое моделирование. Несколько алгоритмов также требуют реализации комплексных классических оракулов. Описание оракула является еще одной важной частью многих квантовых алгоритмов. Оракул, обычно, дается классической функцией, описывающей некоторые аспекты ввода в алгоритм, например, ребер графа, выигрышных позиций, арифметических или теоретико-числовых функций и т.д. Чтобы быть полезными в квантовых вычислениях, оракулы должны быть сделаны обратимыми. Это может быть сделано путем определения функции , таким образом, чтобы функция f : Bool^(n + m) → Bool ^(n + m) определялась как f̀ (x, y) = (x, y ⊕ f (x)). Реверсивная логическая функция f может быть [1,15] унитарной картой, работающей на квантовых битах. В литературе часто описывают оракулов высокого и низкого уровня. Несмотря на то, что оракул манипулирует нетривиальными типами данных (например, целыми числами, действительными числами, ребрами графа, и т.д.), это низкий уровень, так как алгоритм входит в подробности того, как реализовать это в терминах квантовых регистров. Но это также и высокий уровень, в том смысле, что подробности того, как оракул выполняет свои действия, известны пользователю, зачастую, только в общих чертах. Отправной точкой для каждой из реализаций алгоритма было подробное описание алгоритма, представленное IARPA. Многие формализмы программирования квантовых компьютеров были разработаны в течение последних нескольких десятилетий. Некоторые из них, такие как квантовая машина Тьюринга [5] или квантовое лямбда исчисление [6], в основном, теоретические инструменты для изучения конкретных аспектов квантовых вычислений, и не предназначены к практическому квантовому программированию. Есть много недавних предложений квантовых языков программирования [7]. Из них три языка, которые представляют важные вехи, можно рассматривать как предшественников Quipper. Среди императивных языков программирования, пожалуй, старейший квантовый язык программирования QCL [8] определен как язык C- стиля. У QCL есть интересные особенности. Его коллективно окрестили структурированным языком квантового программирования. Это обеспечивает относительно естественный способ написания простых квантовых алгоритмов. Одним из нововведений QCL было разделение функций на отдельные синтаксические классы на основе их поведении во время эксплуатации. Таким образом, QCL отличает неограниченные классические процедуры, квантовые функции ограниченные определением унитарных операций и псевдоклассические операторы, которые предназначены для реализации оракулов, проводящих квантовые испытания и автоматически не вычисляют из ancillas [1,16]. QCL не хватает квантовых типов данных высокого уровня. Он не имеет четко определенной семантики, что осложняет анализ программ. Наконец, поскольку язык был разработан для моделирования, многие из его полезных функций программирования требуют значительных вычислительных мощностей. Несмотря на эти недостатки, QCL является важной вехой в развитии квантовой языков программирования. Совсем недавно предложили два

функциональных квантовых языка программирования, которые могут быть расценены в качестве предшественников Quipper. Квантовое лямбда-исчисление является языком ML- стиля с сильными проверками статического типа [9, 10]. Оно предназначено для работы на модели QRAM [11], но не работает на высоком уровне как средство построения цепи и манипуляции. Квантовая IO Монада [12,13] , как и Quipper, встроенная в Haskell, предоставляет расширяемые типы квантовых данных, и поставляется с последовательной операционной семантикой. Тем не менее, он использует гораздо более простую модель замыкания, и в нем нет многих передовых особенностей программирования в Quipper. Quipper представляет собой масштабируемый функциональный квантовый язык программирования. Его функциональность показало использование семи нетривиальных квантовых алгоритмов, выбранных для представления широкого спектра квантовых вычислительных возможностей. Алгоритмы были реализованы командой из 11 территориально распределенных Quipper программистов. Программирование семи алгоритмов потребовало примерно 55 человеко - месяцев и привело к полезной оценке ресурсов с помощью реалистичных задач. Из этого следует, что Quipper является одновременно удобным и полезным. Одной из задач будущей работы с Quipper будет улучшение проверки типов во время компиляции. Благодаря его Haskell реализации [17], Quipper уже ловит много обычных ошибок типов при компиляции. Тем не менее, во время выполнения, в отсутствие системы линейного типа, определенные свойства, такие как недопущение дублирования квантовых данных, должны быть проверены. Стадия разработки полнофункциональной системы типов будет следующим шагом в развитии Quipper.

1. A. S. Green, P. LeFanu Lumsdaine, N. J. Ross, P. Selinger, B. Valiron. Quipper: A Scalable Quantum Programming Language. http://dl.acm.org/citation.cfm?id=2462177

2. S. Jordan. http://math.nist.gov/quantum/zoo/. Electronic resource.

3. Первый высокоуровневый язык программирования для квантовых компьютеров http://habrahabr.ru/post/185936/

4. IARPA Quantum Computer Science Program. Broad Agency Announcement IARPA-BAA-10-02. Available from https://www.fbo.gov/notices/637e87ac1274d030ce2ab69339ccf93c, April 2010.

5. D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. http://www.cs.berkeley.edu/~christos/classics/Deutsch_quantum_theory.pdf

6. A. van Tonder. A lambda calculus for quantum computation. http://arxiv.org/abs/quant-ph/0307150

7. S. J. Gay. Quantum programming languages: Survey and bibliography. http://www.dcs.gla.ac.uk/scripts/personal/simon/topic

8. B.Omer. Quantum programming in QCL. http://tph.tuwien.ac.at/~oemer/doc/quprog.pdf

9. P. Selinger and B. Valiron. A lambda calculus for quantum computation with classical control. http://www.google.ru/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCkQFjAA&url=http%3A%2F%2Fwww.mscs.dal.ca%2F~selinger%2Fpapers%2Fqlambda.ps&ei=g5EAU4vBHvLZ4QSJxIDAAg&usg=AFQjCNEjirWb0Ie_DjFFMZlXccfA79JO-g&sig2=T-neIiYT0t_605dqYmCLBw&bvm=bv.61535280,d.bGE&cad=rjt

10. P. Selinger and B. Valiron. Quantum lambda calculus. http://www.mscs.dal.ca/~selinger/papers/qlambdabook.pdf

11. E. H. Knill. Conventions for quantum pseudocode.
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.57.1328&rep=rep1&type=pdf

12. T. Altenkirch and A. S. Green. The Quantum IO Monad.
http://ru.scribd.com/doc/166716370/Semantic-Techniques-in-Quantum-Computation

13. М. А. Нильсен, И. Л. Чанг. Квантовые вычисления и квантовая информация. М., Мир, 2006.

14 Gilles Brassard, Peter Høyer, Michele Mosca, Alain Tapp. Quantum Amplitude Amplification and Estimation. http://academic.research.microsoft.com/Publication/3634895/quantum-amplitude-amplification-and-estimation

15. All About Monads. http://www.haskell.org/haskellwiki/All_About_Monads

16. STACS 2003: 20th Annual Symposium on Theoretical Aspects of Computer Science ...
http://books.google.ru/books?id=CDkYBY4EJfUC&pg=PA241&lpg=PA241&dq=uncomputation+from+ancillas&source=bl&ots=m07iqIPXAC&sig=8p3sgWxbDQLMp640yyy9BsIqQ3E&hl=ru&sa=X&ei=TT0CU4S9FIG2yAOE9ICwAQ&ved=0CCYQ6AEwAA#v=onepage&q=uncomputation%20from%20ancillas&f=false

17. The Haskell Programming Language http://www.haskell.org/haskellwiki/Haskell