УДК 519.688

КВАНТОВЫЕ КОМПЬЮТЕРЫ И КВАНТОВЫЕ АЛГОРИТМЫ
Часть 2. КВАНТОВЫЕ АЛГОРИТМЫ

В. М. Соловьев

Соловьев Владимир Михайлович, кандидат технических наук, доцент кафедры математической кибернетики и компьютерных наук, начальник Поволжского регионального центра новых информационных технологий, Саратовский национальный исследовательский государственный университет имени Н. Г. Чернышевского, svm@sgu.ru

В работе рассмотрены принципы построения квантовых алгоритмов и их основные особенности. Показано отличие квантового параллелизма от классических методов высокопроизводительных вычислений. Представлена стратегия разработки квантовых алгоритмов на основе квантовых схем. Предложены методы программирования, реализующие квантовые алгоритмы, с использованием языков высокого уровня. Описан подход, реализации унитарных преобразований, основанный на оракуле.

Ключевые слова: квантовые вычисления, квантовый компьютер, квантовые алгоритмы, кубит, базисные состояния, квантовый гейт, квантовая суперпозиция, квантовая запутанность, квантовый параллелизм, квантовая интерференция, оракул, языки квантового программирования.

DOI: 10.18500/1816-9791-2016-16-1-104-112

Окончание (см. [1]).

ВВЕДЕНИЕ

Как уже отмечалось в первой части [1], современные квантовые технологии могут поддерживать совершенно новые алгоритмы вычислений (квантовые алгоритмы), основанные на принципах квантовой механики, и для их реализации необходимы квантовые компьютеры. Но в настоящее время нет универсального квантового компьютера, а есть только экспериментальные образцы, реализующие отдельные подходы к его созданию. При этом одним из главных направлений деятельности в области квантовых вычислений становится разработка квантовых алгоритмов для решения как классических, так и квантовых задач (в том числе и моделирование квантовых систем). Это один из интенсивных путей развития квантовых вычислений, по которому уже существует более 50 доступных реализаций [2], охватывающих самые широкие области вычислений. В этой связи квантовые вычисления являются крайне интересной и перспективной областью исследований для специалистов в области информационных технологий (IT-специалистов). А для этого нужна более совершенная методическая база для обучения таких IT-специалистов. Она должна быть математически строгой и в то же время как можно проще описывать модели квантовых вычислений, балансируя на грани «ликбеза» по квантовой механике.

1. РАЗРАБОТКА КВАНТОВЫХ АЛГОРИТМОВ

Разработка квантовых алгоритмов отличается от разработки классических алгоритмов, так как парадигма квантовой информатики требует сдвига в сторону парадоксов и «переформатирования» мышления, потому что квантовая механика по своей сути контринтуитивна [3]. Среда разработки (design flow) квантовых алгоритмов должна переводить высокоуровневые квантовые программы в эффективные устойчивые к ошибкам реализации на различных квантовых средах. При этом она должна содержать языки программирования, компиляторы, оптимизаторы, симуляторы, дебаггеры и другие инструменты с хорошо определенными интерфейсами и инкорпорированной устойчивостью к исправлению квантовых ошибок (рис. 1).

Для квантовых компьютеров уже сейчас разработаны языки квантового программирования (табл. 1). Они основываются на языках функционального программирования[1] и реализуют квантовые

[1]Языки функционального программирования (Lisp, Erlang, APL (MatLab), Scala, Miranda, ML, Haskell и т. д.) относятся к декларативным и определяют вычисления как строгие абстрактные понятия и методы символьной обработки данных. В отличие от языков императивного программирования на основе инструкций, изменяющих состояние данных, они не предполагают явного хранения состояния программы. Теоретической моделью этих вычислений является лямбда-исчисление, формализующее понятие вычислимости.

алгоритмы, используя векторную и матричную алгебру. Кроме того, на классических компьютерах можно решать симуляционные задачи, используя фреймворки функциональных языков (например, языка Haskell). Однако реализовать реальные квантовые алгоритмы при помощи таких фреймворков нельзя, поскольку потребуются гигантские вычислительные ресурсы для манипуляции необходимым количеством кубитов и для применения к ним унитарных преобразований. Такие ресурсы не обеспечат даже суперкомпьютеры эксафлопсной производительности, поэтому необходимы именно квантовые компьютеры, которые на физическом (аналоговом) уровне будут выполнять квантовые операции, что значительно эффективнее вычислительной модели на классическом компьютере.

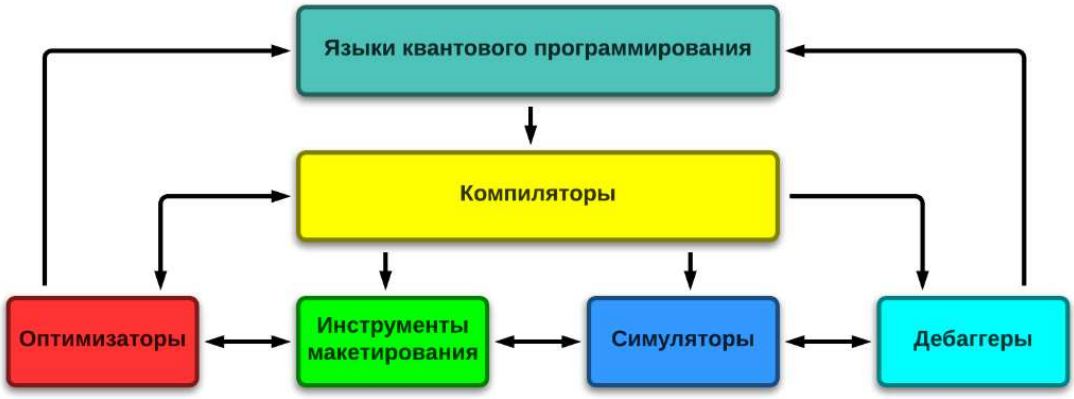

Рис. 1. Среда разработки квантовых алгоритмов

Таблица 1

Языки квантового программирования

Языки QC	Основа языка	Парадигма языка	Адрес в сети	Примечание
QCL	C	ИП	http://www.itp.tuwien.ac.at/~oemer/qcl.html	Первый язык QC
LanQ	Java	ИП	http://lanq.sourceforge.net	Синтаксис языка C
Q-gol	C	ИП	http://www.ifost.org.au/ ~gregb/q-gol	Поддержка графики
Q	C++	ИП	http://q-lang.sourceforge. net	Поддержка прекращена
Pure	C	ИП	http://purelang.bitbucket. org	Приемник языка Q
GCL	C	ИП	http://mirror.tochlab.net/ pub/gnu/gcl	GNU Common Lisp
QPL, QFC		ФП	http://www.vcpc.univie.ac. at/~ian/hotlist/qc/programming.shtml	Языки квантовых схем и симуляторы
QML	Haskell	ФП	http://arxiv.org/abs/quant-ph/0409065	Поддержка графики
Quipper	Haskell	ФП	http://www.mathstat.dal.ca/~selinger/quipper	Последний язык QC
Библиотеки моделирования	C, Java, PHP, Python, и т. д.	–	http://www.quantiki.org/wiki/List-of-QC-simulators	Компьютерная алгебра и основные языки

В настоящее время языки квантового программирования условно можно разделить на два типа: языки, направленные на практическое применение (моделирование квантово-механических систем, программирование квантовых схем и т. д.); языки анализа квантовых алгоритмов. Языки второго типа в основном используются, когда невозможно формально доказать корректность и эффективность алгоритма, так как он может быть основан на недоказанных математических предположениях или эвристических методах. В этом случае чаще всего требуется тестирование алгоритма и его статистический анализ даже без квантового компьютера, используя квантовые виртуальные машины (quantum virtual machine) и библиотеки симуляции квантовых компьютеров. В этом случае, работая с небольшими размерами входных данных, можно осмыслить возможности и проблемы алгоритма.

Это позволяет найти нужные алгоритмы при помощи метода проб и ошибок, а затем перенести их на квантовые компьютеры. Более интересной является архитектура программного обеспечения первого типа, где высокоуровневые языки программирования, позволяют реализовать квантовые алгоритмы (рис. 2).

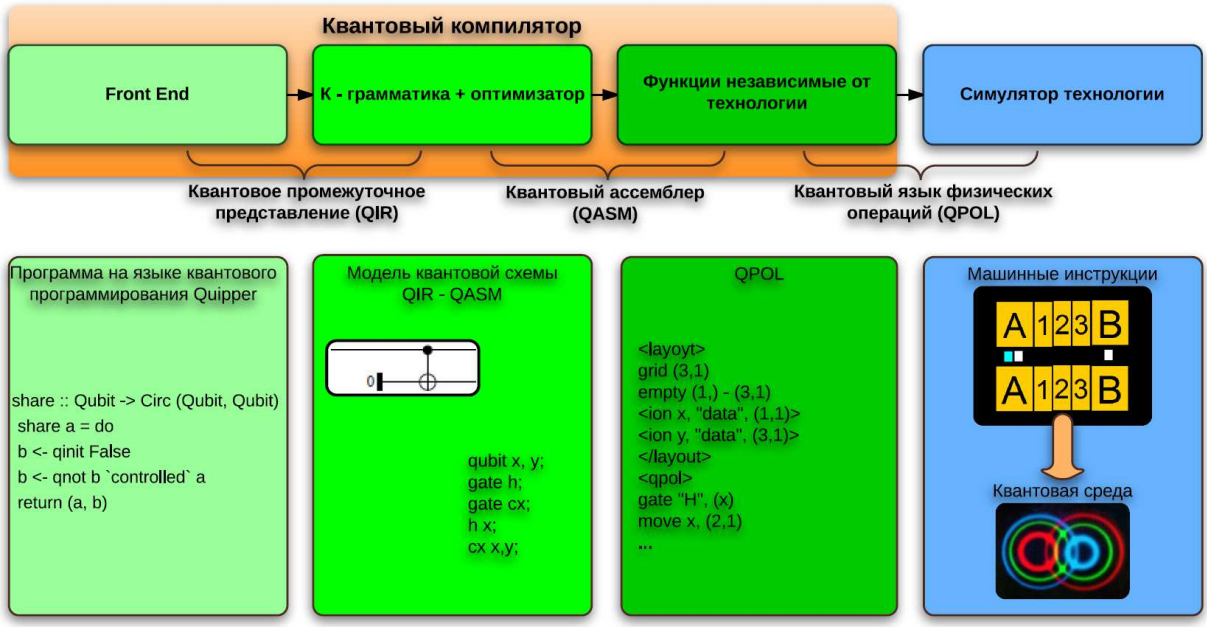

Рис. 2. Программное обеспечение, реализующее квантовые алгоритмы

Алгоритм может быть реализован на независимом от квантовой технологии высокоуровневом языке программирования (например, Quipper). Код программы транслируется фронт-ендом в квантовое промежуточное представление (QIP) на основе квантовой схемы. При этом поддерживается категориальная грамматика (categorial grammatical, CG) и оптимизатор, который выполняет преобразования и оптимизацию кода, независящую от использования квантовых технологий (например, удаление двух последовательных гейтов Адамара). В ходе преобразования QIP описание квантовых схем представляется низкоуровневым описанием на языке квантового ассемблера (QASM). На следующем шаге, уже зависящим от квантовой технологии, оптимизатор преобразует программу на QASM в инструкции квантового языка физических операций (QPOL). Набор этих инструкций QPOL отправляется для вычислений либо квантовому компьютеру, либо симулятору для исполнения. Такая архитектура программного обеспечения позволяет осуществить независимую разработку каждого слоя. Таким образом, разные языки квантового программирования могут использовать различные фронтенды, но один и тот же оптимизатор кода. При этом изменение технологии квантовых вычислений приведёт лишь к замене зависящей от технологии части оптимизатора. Это позволяет независимо проектировать программное обеспечение и гарантировать его интероперабельность.

Для эффективных вычислений в квантовых алгоритмах могут использоваться так называемые оракулы — квантовые аналоги черных ящиков, реализующие, например, унитарные преобразования $U_f : \{0,1\}^{k1+k2} \to \{0,1\}^{k1+k2}$ функции $f : \{0,1\}^{k1} \to \{0,1\}^{k2}$. При этом считается, что существует некоторый физический процесс, вычисляющий обратимым образом эту функцию f, выполняя квантово-механическое унитарное преобразование. В оракуле (рис. 3) k1-кубитный регистр x содержит входные данные для функции f, а k2-кубитный регистр y является вспомогательным (обычно он инициализирован нулями). На выходе результат вычисления функции складывается с этим регистром по модулю 2.

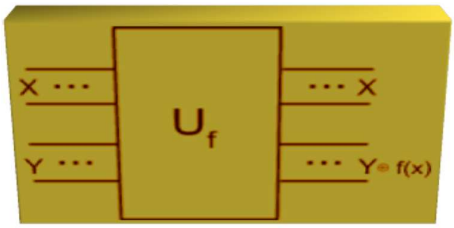

Рис. 3. Оракул U_f

При создании оракула используется метод конструирования коммуникационной квантовой схемы из базисных блоков, реализующей обратимое унитарное преобразование, которое можно использовать в алгоритме.

2. СТРАТЕГИЯ ПОСТРОЕНИЯ КВАНТОВОГО АЛГОРИТМА

Общая стратегия построения квантового алгоритма основана на выборе унитарного оператора преобразования U размерности $2^k \times 2^k$:

$$|\psi_{i+1}\rangle = U(2^k \times 2^k)|\psi_i\rangle \tag{1}$$

и измерениях. В алгоритмах, кроме оператора выражения (1), могут учитываться исходные квантовые состояния и распределения вероятностей. Кроме того, алгоритмы должны исключать копии произвольных квантовых состояний из-за запрета клонирования[2], когда копирование содержимого переменных не допускается в принципе [4]. Квантовая схемотехника создания квантовых алгоритмов — это методология анализа и синтеза схем квантовых вычислений на основе следующей структуры: входные данные, преобразования, выходные данные. В этой связи задачи квантовой схемотехники сводятся к следующему.

1. *Прямой анализ*, когда по схеме входа и описанию вычислительного процесса определяется схема выхода. Математически входная схема — это описание множества возможных значений на входе квантового вычислительного процесса. Поскольку квантовый регистр (набор кубитов) представляется в виде вектора, а каждый гейт в квантовой схеме представляется унитарной матрицей, то задача прямого анализа сводится к последовательному умножению матриц на вектор. Выполнив эту процедуру на всех возможных значениях входа, можно получить все возможные выходные значения, объединив их в схему. Этот процесс затруднителен для классического компьютера, так как при увеличении количества кубитов размерность векторов и матриц растет экспоненциально.

2. *Обратный анализ* в модели квантовых вычислений тривиально сводится к задаче прямого анализа, так как эти вычисления являются обратимыми, а матрицы всех гейтов — унитарными. Поэтому для обратного анализа квантовой схемы достаточно обратить её, то есть перекоммутировать гейты в обратном порядке, сделав выход входом, а вход — выходом, при этом сами гейты преобразовываются в эрмитово-сопряжённые. После всего этого проводится процедура прямого анализа. Это справедливо только для квантовых схем, где нет операций измерения, которые являются необратимыми. Но измерения в большинстве квантовых алгоритмов применяются в конце квантовых вычислений, когда необходимо получить классический результат, а поэтому обращение можно осуществлять, не обращая внимание на измерения. Однако существуют квантовые алгоритмы, в которых измерение производится в середине процесса вычислений (квантовая телепортация). В таких алгоритмах и их квантовых схемах описанный метод обратного анализа использовать нельзя, а нужно использовать иные методы, если они вообще существуют, так как в классических компьютерах задача обратного анализа неразрешима [5].

3. *Синтез квантовой схемы* по заданным входным и выходным данным усложняется по сравнению с классическим компьютером обратимостью вычислений. В общем виде произвольный вычислительный процесс может быть описан как двоичная функция, обрабатывающая входные данные и возвращающая выходные. Такая функция в классическом варианте строится при помощи базисного набора логических элементов. Далее классическая схема, используемая при синтезе, может быть сведена к задаче построения квантового оракула. Однако это не единственное решение задачи синтеза. Другой способ основан на построении одной унитарной матрицы для представления классической функции. Эта задача решается при помощи системы уравнений, получаемой из произведения матрицы на вектор. При этом количество уравнений и неизвестных растёт экспоненциально от количества кубитов (векторов) и решать такую систему на классическом компьютере проблематично.

В общем виде квантовая схема — это только основа для построения квантового алгоритма, которая позволяет решать на квантовом компьютере произвольную вычислительную задачу. Разработка же

[2]Согласно запрета клонирования невозможно создать идеальную копию произвольного неизвестного квантового состояния. Это вытекает из того, что клонирование является операцией, в результате которой создается состояние, являющееся тензорным произведением идентичных состояний подсистем, а идентичности в квантовых системах достичь нельзя.

квантового алгоритма — это дело творческое и нетривиальное, где кроме необходимых знаний и таланта не малую роль играет интуиция. В настоящее время большинство разработанных квантовых алгоритмов (рис. 4) создаются на нескольких базовых, которые именно так и были найдены [6].

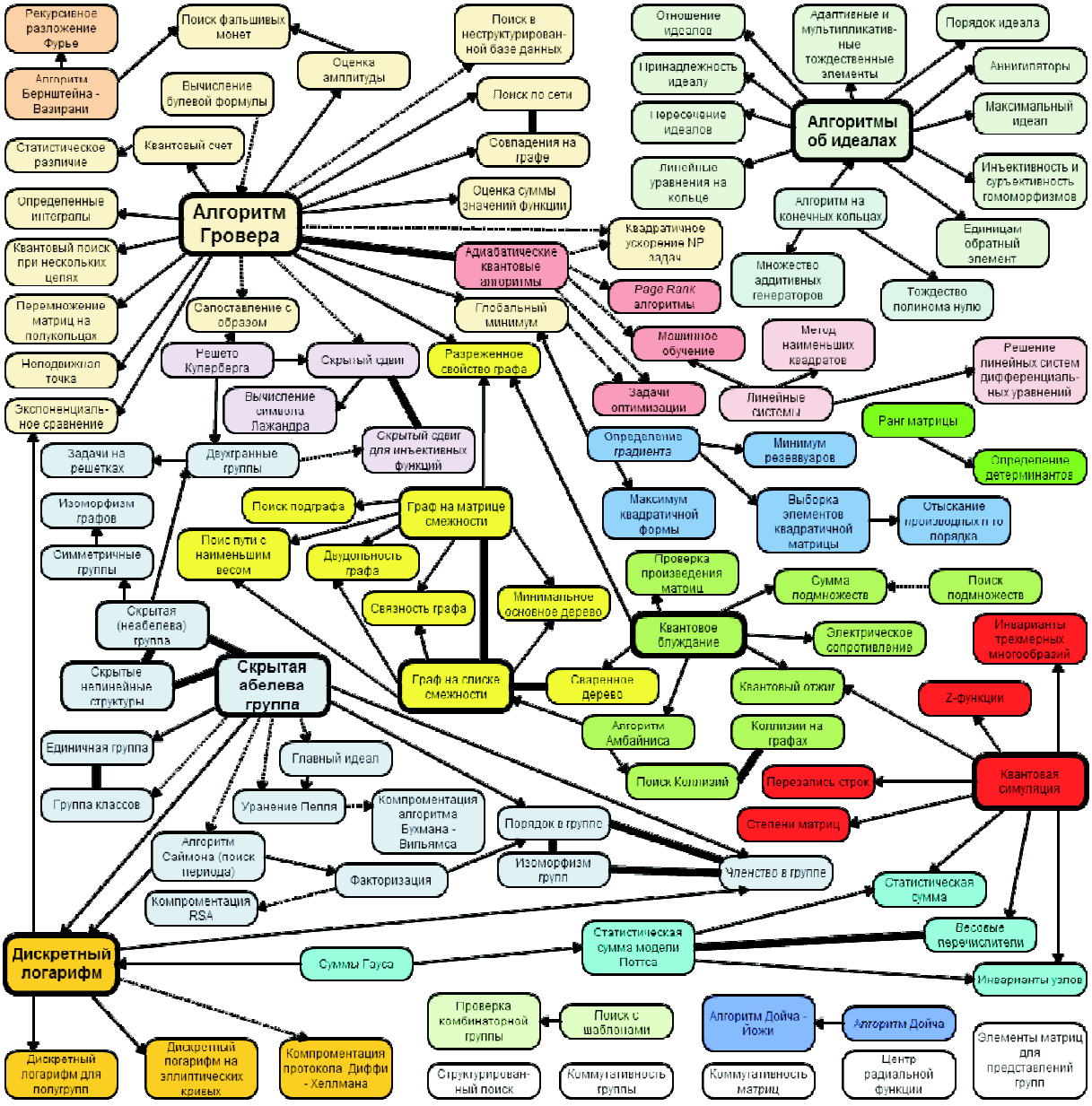

Рис. 4. Диаграмма квантовых алгоритмов

Квантовые алгоритмы отличаются от классических и базируются на следующих принципах.

1. *Обратимость (реверсивность)* вычислений во времени позволяет восстановить исходные данные по результатам вычислений. Для любой квантовой схемы, лежащей в основе алгоритма, можно построить реверсивную схему, которая, получая на входе результат работы первоначальной схемы, возвращает исходные данные для неё. Такую реверсивную схему можно создать, перевернув ее справа налево, а все унитарные преобразования заменить на эрмитово-сопряжённые. Из этого принципа вытекает следующее — квантовые вычисления структурно избыточны[3]. Следовательно, в процессе работы квантового алгоритма будут накапливаться «мусорные» данные, которые нужно удалять, но

[3]Структурная избыточность обычно сводится к лишним (не нужным) входам-выходам и базовым элементам (гейтам).

при этом вычисления станут необратимыми и будет выделяться тепло, что повысит декогеренцию. Однако можно создать реверсивную схему и минимизировать накопление мусора за счет специально разработанных методов уничтожения «мусора» [7]. Обычно они собирают все неиспользованные выходы и преобразуют их специальным образом так, чтобы они использовались (например, для вычисления обратной функции), а весь процесс вычислений и его квантовая схема были полностью реверсивными. В этом случае за экспоненциальное ускорение решения некоторых задач, которое даёт модель квантовых вычислений, придётся заплатить экспоненциальным увеличением размера памяти. Кроме того, обратимость не разрешает в квантовых схемах циклы и возвраты назад. Кубиты как бы двигаются по гейтам, проходя через них и преобразовываясь в соответствии со схемой. Выполнение квантовой программы происходит от начала только вперёд. Единственный способ выполнения алгоритма — унитарные преобразования, а единственный способ получения результата — измерение, уничтожающее суперпозицию, в которой находятся кубиты.

2. *Квантовый параллелизм* обеспечивает параллельное решение одной и той же задачи для экспоненциально больших данных путем прохождения их через гейты унитарных преобразований. Параллелизм обеспечивается суперпозицией базисных состояний кубитов, и с ростом числа кубитов размерность базиса растёт в степенной зависимости. Поэтому квантовый параллелизм обладает огромной вычислительной мощью, которой алгоритмы должны правильно воспользоваться.

3. *Интерференция*, тесно связанная с принципом параллелизма, широко используется в квантовых алгоритмах для взаимного усиления требуемых результатов и ослабления результатов нежелательных. Повторяя несколько раз последовательность параллельной обработки с учетом интерференции состояний кубитов, в алгоритме можно так усилить амплитуду (вероятность) искомого состояния, что в дальнейшем при измерении получится требуемый результат с высокой вероятностью. При этом, варьируя количеством повторений и шагом интерференции, можно управлять вероятностями, доводя их до любого заданного значения.

4. *Квантовая запутанность* — наименее изученный принцип, который не поддается рациональному осмыслению уже в течение века. Но именно этот принцип является ключевым фактором многих квантовых алгоритмов, позволяющим решать неразрешимые классическими компьютерами задачи.

Из вышеперечисленного следует, что не обязательно ждать появления реального универсального квантового компьютера или его облачной реализации (quantum cloud) [8, 9]. Можно уже сейчас, изучая квантовые вычисления, разрабатывать квантовые алгоритмы, делая это поэтапно (рис. 5).

Рис. 5. Этапы разработки квантовых алгоритмов

Во-первых, проанализировать моделируемую физическую систему и ее математическое описание (модель), обратив особое внимание на возможность дискретизации и параллелизма. Тем самым создав вычислительную модель — основу разрабатываемого квантового алгоритма, учитывающего особенности квантовых вычислений. Во-вторых, реализовать будущий квантовый алгоритм на одном из языков функционального программирования (Lisp, Erlang, Scala, Miranda, ML, Haskell) или с помощью систем компьютерной алгебры (Maple, Mathematica, Maxima, MatLab, Octave), симуляторов и соответствующих фреймворков (Eqcs-0.0.8, Q++, QCLib, QCSim, Quantum Computer Simulator, Quantum Network Computing, QC Simulator, QCAD, Quantum Qudit Simulator, QSim, jQuantum, Quantum Algorithm Designer, Quantum eXpress, Haskell Simulator of Quantum Computer и т. д.). В-третьих, используя существующие языки квантового программирования (QCL, LanQ, Q-gol, Q, Pure, GCL, QPL, QML, Quipper), реализовать алгоритмы. При этом нужно учитывать, что размерности используемых в вычислениях векторов (квантовых регистров) и матриц унитарных преобразований (гейтов) растут экспоненциально в зависимости от количества кубитов, используемых в

алгоритме. На современных классических суперкомпьютерах пока можно реализовать квантовые алгоритмы, требующие не более 15 кубитов [7,10]. Современные эмуляторы квантовых алгоритмов, реализованные на классических компьютерах, по сравнению с классическими алгоритмами преимуществ не дают, а для некоторых алгоритмов, использующих небольшие данные, классические алгоритмы работают значительно быстрее. Преимущества квантовых вычислений проявляются только на квантовом компьютере и на больших данных. Поэтому работы по квантовым вычислениям в большинстве своем ведутся пока только в интересах теоретической информатики и фундаментальных исследований. Однако некоторые алгоритмы, приведенные на рис. 4, имеют уже и чисто прикладное значение. Например, алгоритм Шора позволил скомпрометировать криптографический алгоритм RSA и протокол обмена ключами Деффи – Хеллмана, а алгоритмы Гровера, квантового блуждания, нахождения глобального минимума позволили значительно повысить эффективность неструктурированного поиска. Кроме того, фундаментальность и глубина алгоритмов об идеалах, скрытых абелевых групп, дискретных логарифмов дает основание предполагать, что они могут стать основой для решения прикладных задач в ближайшем будущем. Именно такой путь прошли классические вычисления, когда сначала были разработаны алгоритмы с совершенно непонятными структурами данных, а уже потом прикладные программисты реализовывали современное программное обеспечение.

ЗАКЛЮЧЕНИЕ

Квантовые алгоритмы уже сейчас начинают воплощаться в реально функционирующих экспериментальных устройствах, а квантовые вычисления являются довольно развитой областью знаний. В неё вовлечены многие лучшие умы в области физики и информатики [7,11], а количество публикаций растёт день ото дня. Интенсивно создаются новые алгоритмы и способы их применения к решению прикладных задач. Поэтому всем тем, кто хочет заниматься квантовыми вычислениями, уже сегодня надо полноценно погружаться в эту область и изучать её фундаментальные основы.

При создании программного обеспечения (software) квантового компьютера необходимо, прежде всего, решить следующие задачи:

– подготовить алгоритмы унитарных преобразований (решения задачи), исключающие копии произвольных квантовых состояний, циклы и возвраты назад, обеспечивающие сборку и удаление «мусорных» данных, а также коррекцию квантовых ошибок, поддержку квантового параллелизма для экспоненциально больших данных, повторяющие несколько раз параллельную обработку, с учетом интерференции состояний кубитов и повышающие вероятность результата;

– написать и отладить программу на языке функционального или квантового программирования, реализующую подготовленный алгоритм (подготовку и отладку программ можно выполнять и на классическом компьютере, компиляторы которого совместимы с компиляторами квантового компьютера, кроме того, необходимо учитывать ограничения на размеры обрабатываемых данных и наличие «драйверов» для низкоуровневого доступа к квантовому компьютеру);

– предусмотреть вывод результата на классическом компьютере, работающем как элемент квантового компьютера или с использованием квантовых облачных технологий.

Современный этап развития квантовых вычислений является этапом фундаментальных исследований и экспериментального подтверждения результатов этих исследований (табл. 2). Этот этап позволит выбрать из нескольких прототипов квантовых компьютеров, реализуемых по разным технологиям создания квантовой среды, лучший. Его преимущества будут проявляться, прежде всего, в эффективности решения вышеперечисленных проблем.

Собственно говоря, квантовый компьютер ничего не будет вычислять в обычном смысле. Он как бы заранее будет знать все возможные решения. Останется только отбросить неверные результаты посредством квантовых алгоритмов. Кроме того, на этом этапе необходимо будет решать, как задачи создания программного обеспечения, так и задачи подготовки специалистов в области квантовых вычислений. Этим специалистам придется создавать образцы пока трудно реализуемой гипотетической аналогово-цифровой вычислительной системы, создавать новые и использовать уже появляющиеся квантовые алгоритмы.

Таблица 2

Исследования по созданию квантового компьютера

Компании	Квантовая среда	Особенности
IBM	Исследования квантовой среды на основе схем из сверхпроводящих металлов	Очень высокая вероятность квантовых ошибок, что не позволяет создавать полноценные квантовые компьютеры
Microsoft	Исследование теоретически более надежной квантовой среды и создание топологического кубита*	Существование квазичастиц, используемых в топологическом кубите, пока не доказано
Alcatel-Lucent (Bell Labs)	Исследования конденсированного состояния вещества с целью создания топологического кубита	Создание топологического кубита на основе дробного квантового эффекта Холла пока в стадии исследований
D-Wave Systems	Исследования по созданию квантового компьютера на основе сверхпроводящего чипа, содержащего 512 кубитов	Пока не доказано, что чипы построены на основе квантовых эффектов
Google	Разноплановые исследования компьютеров D-Wave Systems, построенных на основе контактов Джозефсона	Google адаптирует свои технологии под возможности квантовых компьютеров

Примечание. *Топологический кубит — это теоретический кубит на основе двухмерных квазичастиц (анионов), являющихся более стабильными, что позволяет уменьшить ошибки декогеренции. Это одно из трех направлений уменьшения ошибок в квантовых компьютерах (первое — коррекция, второе — подавление декогеренции). Топологическое состояние анионов подразумевает неизменность топологии при изменении их состояния, базирующейсе на принципе запрета Паули (две частицы не могут находиться в одинаковом состоянии). Состояния из нескольких анионов соответствуют переплетению топологий, аналогично пряже. Это позволяет построить математическую теорию соответствующих групп и алгебр, называемых не абелевыми. На этих состояниях в результате переплетения, можно построить квантовый компьютер.

Библиографический список

1. *Соловьев В. М.* Квантовые компьютеры и квантовые алгоритмы. Ч. 1. Квантовые компьютеры // Изв. Сарат. ун-та. Нов. сер. Сер. Математика. Механика. Информатика. 2015. Т. 15, вып. 4. С. 462–477. DOI: 10.18500/1816-9791-2015-15-4-462-477.
2. Algebraic and Number Theoretic Algorithms. URL: http://math.nist.gov/ quantum/zoo/ (дата обращения: 23.06.2015).
3. *Богданов Ю. И., Кокин А. А., Лукичёв В. Ф., Орликовский А. А., Семенихин И. А., Черняв-ский А. Ю.* Квантовая механика и развитие информационных технологий // Информационные технологии и вычислительные системы. 2012. № 1. С. 17–31.
4. *Venegas-Andraca S. E.* Quantum Walks for Computer Scientists. Synthesis Lectures on Quantum Computing. Morgan Claypool, 2008. 133 p.
5. *Горбачев В. Н., Жилиба А. И.* Физические основы современных информационных процессов или учебное пособие по квантовой телепортации, квантовым вычислениям и другим вопросам квантовой информации. Тверь : Из-во Твер. гос. ун-та, 2001. 43 с.
6. *Shor P. W.* Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. arXiv: quant-ph/9508027. 25.01.1996. 28 p.
7. *Williams C. P.* Explorations in Quantum Computing. Springer-Verlag London Ltd., 2011. 740 p.
8. Closing in on quantum computing. URL: http:// www.wired.com/2014/10/quantum-computing-close (дата обращения: 23.06.2015).
9. *Bhambri S.* Quantum Clouds: A future perspective. arXiv: quant-ph/1410.6502v1. 05.10.2014. 14 p.
10. *Валиев К. А.* Квантовые компьютеры и квантовые вычисления // Успехи физических наук. 2005. Т. 175, № 1. С. 3–39.
11. *Metodi T. S., Faruque A. I., Chong F. T.* Quantum Computing for Computer Architects. Synthesis Lectures on Computer Architecture. Morgan Claypool, 2011. 203 p.

Quantum Computers and Quantum Algorithms. Part 2. Quantum Algorithms

V. M. Solovyev

Solovyev Vladimir Mihajlovich, Saratov State University, 83, Astrakhanskaya st., Saratov, Russia, 410012, svm@sgu.ru

The paper discusses principles of construction for quantum algorithms and their main features. Distinction of quantum parallelism from classical methods of high-performance computing is shown. Quantum algorithms design strategy is presented based on quantum circuits. Methods of programming for implementation of quantum algorithms using high-level languages are proposed. An approach to implement unitary transformations based on the oracle method is described.

Key words: quantum computing, quantum computers, quantum algorithms, qubit, quantum gate, quantum superposition, quantum entanglement, quantum parallelism, quantum interference, oracle, quantum-programming languages.

References

1. Solovyev V. M. Quantum Computers and Quantum Algorithms. Pt. 1 : Quantum Computers. *Izv. Saratov Univ. (N.S.), Ser. Math. Mech. Inform.*, 2015, vol. 15, iss. 4, pp. 462–477. DOI: 10.18500/1816-9791-2015-15-4-462-477 (in Russian).
2. *Algebraic and Number Theoretic Algorithms*. Available at: http://math.nist.gov/ quantum/zoo/ (accessed 23 June 2015).
3. Bogdanov U. I., Kokin A. A., Lukichev V. F., Orlikovskij A. A., Semenihin I. A., Chernavskij A. U. Quantum mechanics and the development of information technology. *Information technologies and computer systems*, 2012, no. 1, pp. 17–31 (in Russian).
4. Venegas-Andraca S. E. *Quantum Walks for Computer Scientists*. Synthesis Lectures on Quantum Computing, Morgan Claypool, 2008, 133 p.
5. Gorbachev V. N., Zhiliba A. I. *Physical basis of modern information processes or textbook on quantum teleportation, quantum computing and other issues of quantum information*. Tver, Tver State University, 2001, 43 p. (in Russian).
6. Shor P. W. *Polynomial-time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*. arXiv: quant-ph/9508027, 25.01.1996, 28 p.
7. Williams C. P. *Explorations in Quantum Computing*. Springer-Verlag London Ltd., 2011, 740 p.
8. *Closing in on quantum computing*. Available at: http://www.wired.com/2014/10/quantum-computing-close (accessed 23 June 2015).
9. Bhambri S. *Quantum Clouds: A future perspective*. arXiv: quant-ph/1410.6502v1, 05.10.2014, 14 p.
10. Valiev K. A. Quantum computers and quantum computing. *Uspekhi Fizicheskikh Nauk* [Successes of physical sciences], 2005, vol. 175, no 1, pp. 3–39 (in Russian).
11. Metodi T. S., Faruque A. I., Chong F. T. *Quantum Computing for Computer Architects*. Synthesis Lectures on Computer Architecture, Morgan Claypool, 2011, 203 p.

УДК 517.11

ОБ ОДНОМ ПОДХОДЕ К НЕЧЕТКОМУ ЛОГИЧЕСКОМУ МОДЕЛИРОВАНИЮ ЦИФРОВЫХ УСТРОЙСТВ

Д. В. Сперанский

Сперанский Дмитрий Васильевич, доктор технических наук, профессор кафедры высшей и прикладной математики, Московский государственный университет путей сообщения, Speranskiy.dv@gmail.com

В статье исследуется проблема двоичного нечеткого моделирования цифровых устройств (ЦУ). В отличие от аналогичной классической проблемы предполагается, что входные сигналы ЦУ являются нечеткими. В реальных ЦУ для каждого входа (0 или 1) существует определенный диапазон в вольтах. Если входной сигнал выходит за этот диапазон, то корректность его идентификации не гарантируется. Нечеткость входного сигнала означает, что наблюдаемые его значения могут быть либо внутри определенного диапазона, или вне его. Известно, что логическое моделирование каждого ЦУ состоит в вычислении значения определенного логического выражения. Это выражение есть математическая модель ЦУ. Кроме того, это логическое выражение может всегда быть представлено в терминах трех логических операций, а именно И, ИЛИ, НЕ. В статье предлагается метод сведения исследуемой проблемы к проблеме нечеткого моделирования систем в пространстве вещественных чисел. Метод основан на представлении логического выражения с исполь-