Научная статья
УДК 519.688:004.4
ББК 22.18+32.973.2
Г 27
doi: 10.53598/2410-3225-2021-2-281-56-65

Разработка и исследование программного обеспечения и методов построения квантовых алгоритмов для решения задач NP класса

(*Рецензирована*)

Сергей Михайлович Гушанский[1], Виктор Сергеевич Потапов[2], Максим Сергеевич Коваленко[3]

[1, 2, 3] *Южный федеральный университет, Таганрог, Россия*
[1] *smgushanskiy@sfedu.ru*
[2] *vitya-potapov@rambler.ru*
[3] *kovalenko.max61@gmail.com*

Аннотация. *В современных науке и технике постоянно возникает необходимость в решении таких стратегически важных задач, как предсказание погоды и расчет климатических изменений, создание онкологических препаратов, обработка сигналов из Вселенной для поиска внеземных цивилизаций, обработка символьной информации, криптоанализ, опережающий расчет траекторий движущихся воздушных и космических объектов и другие задачи. Практическая реализация перечисленных задач на современных, даже суперкомпьютерных, системах требует недопустимо большого промежутка времени или вообще невозможна. Научная новизна данного направления выражается в разработке методики построения модульной моделирующей системы квантовых вычислений с открытой архитектурой, а также моделирующей системы выполнения квантовых алгоритмов. Целью работы является рассмотрение приемов моделирования квантовых вычислений и реализации алгоритмов решения задач NP класса.*

Ключевые слова: *квантовый алгоритм, запутанность, кубит, квантовый гейт, квантовый симулятор*

Original Research Paper

Development and research of software and methods for constructing quantum algorithms for solving NP class problems

Sergey M. Gushansky[1], Viktor S. Potapov[2], Maksim S. Kovalenko[3]

[1, 2, 3] *Southern Federal University, Taganrog, Russia*
[1] *smgushanskiy@sfedu.ru*
[2] *vitya-potapov@rambler.ru*
[3] *kovalenko.max61@gmail.com*

Abstract. *In modern science and technology, there is a constant need to solve such strategically important tasks as predicting the weather and calculating climatic changes, creating cancer drugs, processing signals from the Universe to search for extraterrestrial civilizations, processing symbolic information, cryptanalysis, and anticipatory calculation of the trajectories of moving air and space objects, etc. The practical implementation of the listed tasks on modern, even supercomputer systems requires an unacceptably long period of time or is impossible at all. The scientific novelty of this direction is expressed in the development of a methodology for constructing a modular modeling system of quantum computing with an open architecture, as well as a modeling system for executing quantum algorithms. The aim of the work is to consider techniques for modeling quantum computations and implementing algorithms for solving NP class problems.*

Keywords: *quantum algorithm, entanglement, qubit, quantum gate, quantum simulator*

Введение

Квантовые вычисления [1] в настоящее время находятся в эре шумных квантов промежуточного масштаба, когда устройства, состоящие из нескольких кубитов [2], могут выполнять нетривиальные квантовые вычисления, борясь с немалым шумом и отсутствием коррекции ошибок. Так называемые гибридные квантово-классические алгоритмы [3] и, в частности, вариационные квантовые алгоритмы составляют одно из наиболее подходящих подмножеств алгоритмов. Такие алгоритмы используют классические сопроцессоры для итеративного повышения производительности параметризованных квантовых схем по отношению к множеству задач. Вариационные квантовые алгоритмы обычно имеют небольшую глубину и обладают большей устойчивостью к шуму, что необходимо для извлечения потенциала краткосрочных устройств. Такие алгоритмы, как вариационный квантовый вычислитель собственных чисел и алгоритм квантовой приближенной оптимизации, показали большие перспективы в решении сложных проблем в различных полях, включая квантовую химию, материаловедение, финансы и квантовое машинное обучение [4]. Быстрому распространению алгоритмов помогает постоянно расширяющийся рынок пакетов квантовой информации и моделирования.

1. Разработка квантовой вычислительной среды

Разработанная в рамках исследований квантовая вычислительная среда (quantum calculation system – QCS) отличается от аналогов различием в функциональности, а также интерфейсом прикладного программирования (API). QCS – это программный пакет python3 с открытым исходным кодом, который объединяет разнообразное программное обеспечение для моделирования, классические процедуры оптимизации и мощные инструменты для манипуляции и комбинирования квантовых схем [5] и вариативные цели. QCS поддерживает строго объектно-ориентированный пользовательский интерфейс, в котором схемы, гамильтонианы, ожидаемые значения и определяемые пользователем цели могут быть удобно объединены арифметически, используя код, который воспроизводит лежащую в основе математику в виде «доски». Опишем примеры того, как создание, компиляция, манипулирование, дифференциация и оптимизация вариационных квантовых целей могут быть выполнены интерактивно, чтобы проиллюстрировать ядро интерфейса прикладного программирования. Опишем API-интерфейс и то, как он работает с абстрактными структурами данных в форме целей, гамильтонианов [6] и квантовых схем, а также с моделированием и выполнением этих структур. Основная цель QCS – предоставить среду с открытым исходным кодом для быстрой разработки и демонстрации новых идей в квантовых вычислениях с акцентом на вариационные алгоритмы.

Общий интерфейс разработанной модели представлен на рисунке 1. Сверху расположены кнопки управления квантовой схемой. Данная область программы дает возможности автоматического или пошагового хода работы модели вперед или в обратную сторону, также можно удалить последний выбранный и занесенный в схему элемент или полностью очистить всю схему.

Здесь находится полоса меню для управления и конфигурирования модели, по центру сверху располагается набор квантовых гейт [7], снизу диаграмма состояний *x*-регистров и *y*-регистров. Подробно рассмотрение особенности каждого компонента интерфейса пользователя, а также их возможности представлены ниже. Квантовая схема генерируется и обновляется автоматически после добавления нового гейта в схему. QCS спроектирован с использованием высокого уровня абстракции. Пользователю предлагается выбор, проиллюстрированный на рисунке 1, между обработкой квантовых компьютеров как семплерами абстрактных математических ожиданий или более непо-

средственным контролем использования квантового компьютера.

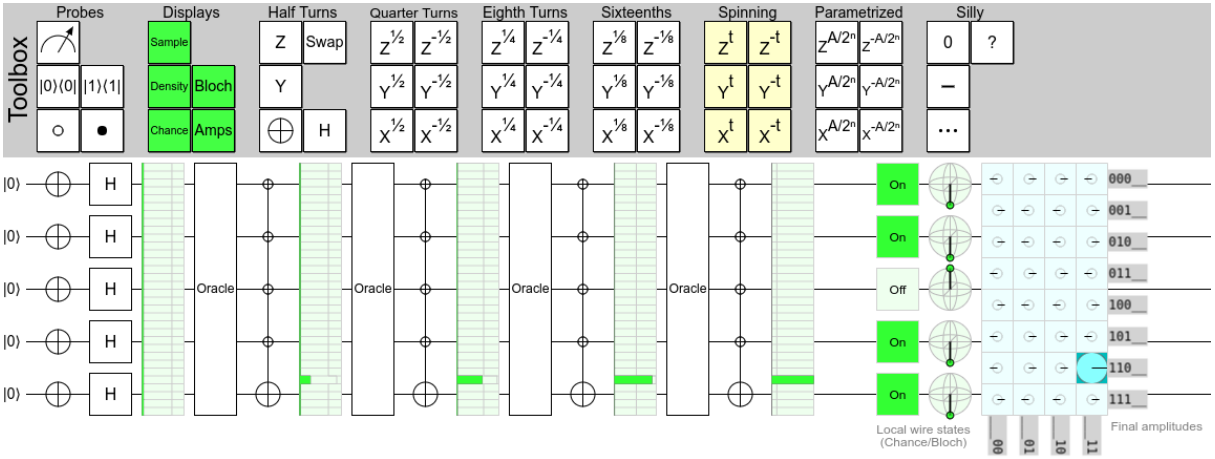

Рис. 1. Интерфейс разработанной среды моделирования

Теория абстрактных структур данных позволяет пользователям рассматривать квантовые серверы, которые служат интерфейсами к реальному оборудованию или симуляторам, как абстрактные устройства выборки, не требующие особых знаний о лежащих в основе технических деталях моделирования или выполнения. Объекты – это вызываемые структуры данных, которые включают список, содержащий абстрактные ожидаемые значения и переменные, а также преобразование, которое определяет, как эти структуры должны обрабатываться после оценки. Формально эквивалентную цель можно записать как $O=f(E0,E1,...,a0,a1,...)$, с переменными a и математическими ожиданиями E. Эти ожидаемые значения также зависят от переменных через унитарные схемы U. Возьмем, например, следующий гамильтониан, действующий на три кубита (указанные в скобках):

$$H = \sigma_x(0)\sigma_y(1) + 3\sigma_y(3).$$

Он может быть инициализирован путем объединения примитивов как:

*H = tq.paulis.X(0) * tq.paulis.Y(1)*
*H += 3.0 * tq.paulis.Y(3)*
*H = tq.QubitHamiltonian.from_string ("1.0 * X(0) * Y(1) + 3.0 * Y(3)")*

Абстрактные квантовые схемы могут быть определены над примитивными квантовыми вентилями либо из тех, что входят в набор, либо путем их определения над генератором, который сам задается гамильтонианом. В качестве примера рассмотрим их вращение на кубите 0

$$R_y(a) = e^{-\frac{a}{2}\sigma_y(0)},$$

которое может быть инициализировано следующими эквивалентными способами:

U = tq.gates.Ry(angle = 1.0, target = 0)
U = tq.gates.ExpPauli(angle = 1.0, paulistring = "Y (0)")
g = tq.paulis.Y(0)
U = tq.gates.Trotterized(angles = [1.0], generators = [g], steps = 1)

Функция троттеризации [8] принимает произвольные эрмитовские генераторы и следует тем же соглашениям, что и однокубитовые вращения [9]. В этом примере не используется приближение. Один для состояний, которые известны аналитически, и один для состояний, закодированных в квантовых схемах. Используем одинаковые состояния Белла [10] для обеих иллюстраций. Точность вычисляется относительно состо-

яния, закодированного в схеме U, которая, как предполагается, уже инициализирована. Здесь точен единственный шаг формальной троттеризации; однако это не так для всех генераторов. В общем случае допускается определение произвольных эрмитовых генераторов гейтов

$$G = \sum_k c_k \sigma_k \ ,$$

представленных взвешенной суммой по k строкам Паули [11] yk, как формальное разложение Троттера с N шагами. Параметризованные гейты, такие как вращения, также могут быть инициализированы переменными или преобразованиями переменных. Квантовые схемы обычно состоят из более чем одного гейта, и их построение может быть достигнуто простым сложением отдельных вентилей вместе. Здесь арифметика реализована в соответствии с моделью квантовой схемы, где сложение интерпретируется как объединение двух схем, а самая левая схема – та, которая действует первой. На верхнем уровне QCS не различает гейты и схемы. Вышеуказанные вращения инициализируются как схемы, содержащие один вентиль. Схемы можно оценивать так же, как и цели, если кто-то интересуется моделированием волновой функции или выборкой из ее распределения, и аналогичным образом с функцией компиляции – wfn=tq.simulate (U,variables={"a": 1.0}). Тип возвращаемых данных одинаков для конечной выборки и явного моделирования, где первый содержит счетчики для соответствующих измерений, а второй – смоделированные амплитуды [12].

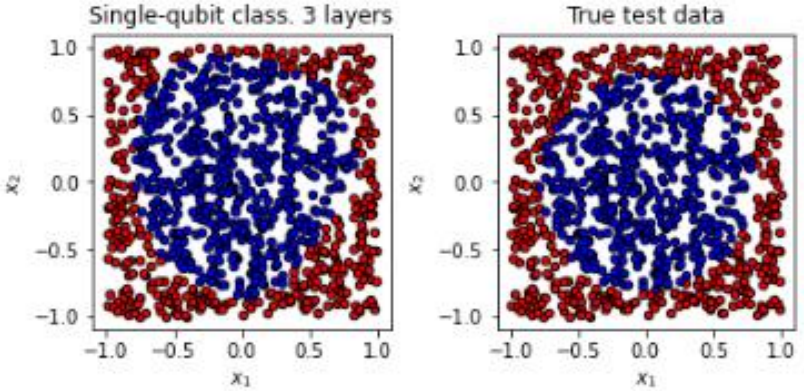

Рис. 2. Результаты после запуска трехслойного однокубитного классификатора с использованием QCS. Используемый оптимизатор – RMS с 400 и 1000 тренировочными и тестовыми точками соответственно. Достигнутая точность составляет 90,5%

2. Реализация квантового алгоритма для решения задачи коммивояжера

Задача коммивояжера – это комбинация набора городов и расстояния между каждыми двумя городами, чтобы найти тот, у которого наименьшее общее расстояние, пройденное туром, который объезжает все города только один раз и возвращается к месту отправления. Эта проблема относится к классу трудностей NP в теории вычислительной сложности. Квантовый отжиг – это метаэвристика для нахождения глобального минимума заданной целевой функции по заданному набору возможных решений с помощью процесса, использующего квантовые флуктуации. Квантовый метод Монте-Карло [13] используется для моделирования квантового отжига. Для воспроизводимости результатов работы алгоритма и его выполнения устанавливаем случайные точки-маршруты коммивояжера. Задаем координаты города для x и y, чтобы решить задачу коммивояжера. Длина x и y должна быть одинаковой. В качестве примера рассмотрим поездки по муниципалитетам в префектуре Акита в Японии. В данных широты и долготы 25 точек. Похоже, что суперкомпьютер K рассчитывает кратчайший маршрут

с использованием грубой силы, что занимает 359 дней. Широта хранится в Аките [, 1]. Долгота хранится в Аките [, 2]. Выполним квантовый отжиг с помощью функции qatsp-result<-qatsp (*x*=Акита [, 1], *y*=Акита [, 2]). По умолчанию trace=TRUE, а переход на кратчайшее расстояние во время расчета отображается на графике. При желании, установив route=TRUE, кратчайший маршрут отображается каждый раз, когда кратчайший маршрут обновляется во время расчета. Другими параметрами функции qatsp являются бета=50, trotter=10, ann_para=1, ann_step=500, mc_step=5000 и reduc=0,99 по умолчанию. Функция сводки может отображать сводку – summary (result). Отображаются кратчайший путь, кратчайшее расстояние, параметры, используемые для расчета, и время расчета. Переход кратчайшего расстояния отображается функцией построения графика – plot (result). Кратчайший путь (рис. 3), полученный функцией маршрута, отображается – route (result).

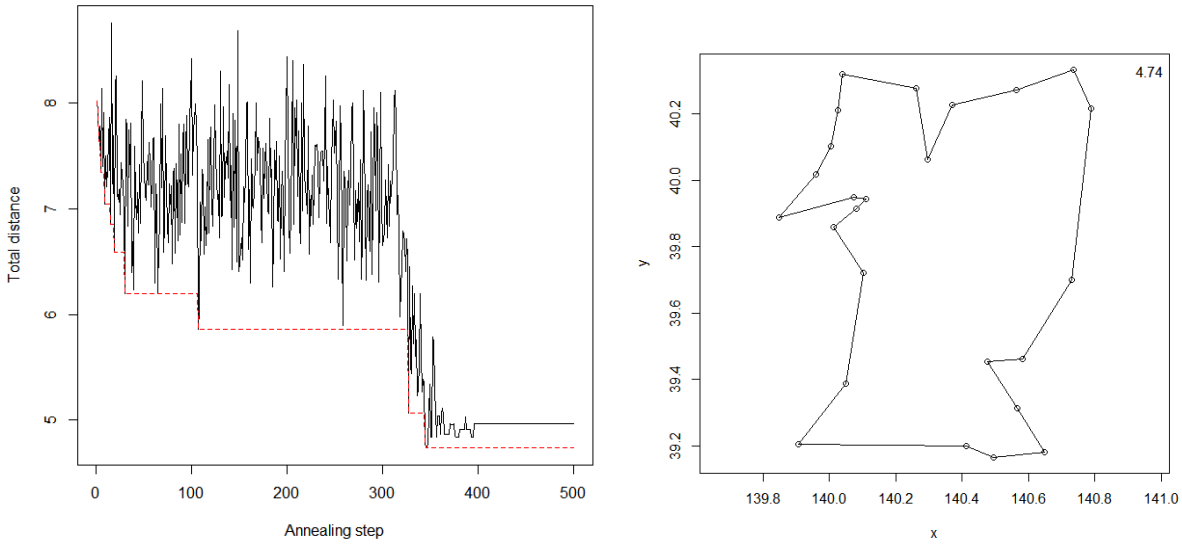

Рис. 3. Шаг отжига против общего расстояния (слева);
кратчайший путь, полученный функцией маршрута (справа)

3. Метод построения квантовых алгоритмов для решения задач NP класса

a. Инициализация (также известная как подготовка состояния)

Цель: в начале квантового алгоритма квантовый регистр [14], которым управляет алгоритм, должен быть инициализирован. Инициализация должна быть максимально простой с учетом требований этапов алгоритма. Алгоритм обычно требует ввода, представляющего параметры решаемой проблемы. Большинство квантовых алгоритмов кодируют этот ввод как часть унитарных преобразований, составляющих квантовый алгоритм. Например, если общий алгоритм $U = U_n \circ ... \circ U_i \circ U_{i-1} \circ ... \circ U_1$, то $U_1,...,U_{i-1}$ – это операторы, которые предоставляют регистр для хранения параметров задачи, решаемой с помощью следующих операторов $U_i,...,U_n$. Однако начальное состояние должно быть установлено, которое называется подготовкой квантового состояния. *Решение:* часто регистр инициализируется как единичный вектор |0...0>. В этом регистре могут быть выделены определенные вспомогательные биты или биты рабочего пространства, которые используются для хранения промежуточных результатов, для управления обработкой алгоритма и т.д. Например, регистр инициализируется с помощью $|0\rangle^{\otimes_n}|0\rangle^{\otimes_m}$ (где вторая часть регистра состоит из битов рабочего пространства) для вычисления таблицы функций булевой функции $f:\{0,1\}^n \to \{0,1\}^m$. Инициализация с помощью

$|0\rangle^{\otimes_n}|1\rangle$ поддерживает выявление принадлежности к набору, который определяется на основе индикаторной функции (например, используемой для решения задач принятия решений) путем изменения знака кубитов, представляющих элементы этого набора. На основе этих простых инициализаций можно подготовить более сложные состояния [15]. Например, в [16] обсуждается несколько алгоритмов загрузки классических битов в квантовый регистр. [17] представляет, как загрузить комплексный вектор, [18] – как загрузить действительный вектор на основе соответствующих структур данных. Все алгоритмы должны быть как-то инициализированы. Часто после инициализации регистр должен быть приведен в состояние однородной суперпозиции. Для функциональных таблиц требуются описанные здесь инициализации. Инициализированный регистр может стать входом для оракула.

b. Единая суперпозиция

Цель: как правило, отдельные кубиты квантового регистра должны находиться в нескольких состояниях одновременно, не отдавая предпочтение ни одному из этих состояний в начале вычислений. Одна из причин силы квантовых алгоритмов проистекает из квантового параллелизма, то есть способности квантового регистра отображать несколько значений одновременно. Это достигается путем совмещения (подмножества) кубитов квантового регистра в суперпозиции. Многие алгоритмы предполагают, что вначале эта суперпозиция однородна, то есть вероятность измерения любого из кубитов одинакова. *Решение:* равномерная суперпозиция достигается за счет инициализации квантового регистра как единичного вектора $|0\ldots0>$ и последующего применения преобразования Адамара:

$$H^{\otimes n}(|0\rangle^{\otimes_n}) = \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle .$$

В случае если квантовый регистр включает в себя вспомогательные биты или биты рабочего пространства в дополнение к вычислительной базе, вычислительная база приводится в суперпозицию. Остальные биты могут быть совмещены или нет. Это достигается с помощью тензорного произведения $H^{\otimes n}\otimes U$, где $H^{\otimes n}$ работает на вычислительной основе, а U работает с другими битами. Создание однородной суперпозиции использует инициализацию. Регистр в равномерной суперпозиции может быть запутанным. Регистр в равномерной суперпозиции может быть введен в оракул.

c. Создание запутанности

Цель: сильная корреляция между кубитами квантового регистра часто требуется, чтобы включить алгоритмы, которые предлагают ускорение по сравнению с классическими алгоритмами. Запутанность – одна из причин силы квантовых алгоритмов, хотя запутанность не является необходимостью. Таким образом, после инициализации квантовый регистр часто приходится запутывать для дальнейшей обработки. *Решение:* для создания запутанного состояния можно использовать несколько подходов.

d. Таблица функций

Цель: некоторые проблемы можно свести к определению глобальных свойств объекта функция. Для этого необходимо вычислить соответствующую таблицу функций. Для вычисления таблицы функций классический алгоритм требует вызова функции для каждого значения домена. *Решение:* квантовый регистр разбивается на вычислительную основу (область функция f), состоящую из n кубитов x, и рабочее пространство, состоящее из m кубитов y, которое используется для хранения значений f. Исходя из

этого, унитарный оператор $U|x,\ y\rangle=|x,\ y\oplus f(x)\rangle$. После инициализации регистра приводится в однородную суперпозицию, оставляя рабочее пространство неизменным, а затем оператор U применяется только один раз, в результате чего получается таблица функций. В случае индикаторной функции f (например, если f представляет проблему решения), регистр инициализируется с помощью $|0\rangle^{\otimes n}|1\rangle$. Равномерное наложение полного регистра предоставлен $H^{\otimes n+1}$. Наконец, применение U приводит к

$$|0\rangle^{\otimes n}|1\rangle \mapsto (\frac{1}{\sqrt{2}}\sum_{x}|x\rangle)\otimes|-\rangle \mapsto= (\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle)\otimes|-\rangle .$$

Таким образом, члены вычислительного базиса своим знаком указывают, являются ли они функцией индикатора (знак минус) или нет (знак плюс) – также называемая «фаза отката». Функциональные таблицы требуют инициализации, о которой говорилось ранее. Равномерная суперпозиция вычислительной базы устанавливается до того, как вычисляется таблица функций.

e. Оракул (черный ящик)

Цель: квантовым алгоритмам часто требуется вычислять значения функции f без необходимости знания подробностей того, как вычисляются такие значения. Концепция оракула (или черного ящика) как гранула повторного использования со скрытыми внутренними компонентами поддерживает этот метод построения квантовых алгоритмов. *Решение:* оракулы используются для решения конкретных задач. [19] обсуждает различные виды оракулов. Ограничения использования оракулов обсуждаются в [20]. Оракул часто требует не вычислять состояние своего результата и предполагает, что подготовленный регистр – как вход (инициализация).

f. Развычисление (обратное вычисление)

Цель: часто запутывание вычислительной основы квантового регистра с временными кубитами (вспомогательная, рабочая область) должны быть удалены, чтобы обеспечить правильное продолжение алгоритма. Для вычисления часто требуются временные кубиты в конце вычисления. Эти кубиты связаны с вычислительной базой, что мешает доступу к фактическому результату вычисления, особенно если вычисление использовалось просто как промежуточный шаг в алгоритме. *Решение:* большинство алгоритмов преобразуют $|x\rangle|0\rangle|0\rangle$ в $|x\rangle|g(x)\rangle|f(x)\rangle$ для вычисления функции f. То есть вторые кубиты представляют собой рабочую область, которая содержит мусор $g(x)$ в конце вычисления.

g. Сдвиг фазы

Цель: в данном регистре следует выделить определенные кубиты. Когда алгоритм применяется итеративно и предполагается, что каждая итерация улучшит решение, следует указать те части решения, которые действительно улучшились. Таким показателем может быть фазовый сдвиг. *Решение:* следующий оператор S может быть эффективно реализован по количеству используемых гейт:

$$\sum_{x=0}^{N-1}a_x|x\rangle \mapsto \sum_{x\in G}e^{i\varphi}a_x|x\rangle+\sum_{x\notin G}a_x|x\rangle .$$

Этот оператор сдвигает кубиты в $G\subseteq\{0,\ldots,N\text{-}1\}$ на фазы φ и оставляет остальные кубиты без изменений. Есть даже вариант оператора, который сдвигает фазы кубитов в хорошем наборе на разные значения, то есть $\varphi=\varphi(x)$.

h. Амплитудное усиление

Цель: на основе приближенного решения вероятность найти точное решение следует увеличивать от запуска к запуску алгоритма U. Таблица функций индикаторной функции f может перечислять все решения проблемы (то есть $f(x)=1 \Leftrightarrow x$ решает проблему). Измеряя соответствующее состояние, решение будет найдено с определенной вероятностью. *Решение:* состояние преобразуется таким образом, что интересующие значения амплитуды изменяются так, что они получают более высокую вероятность быть измеренными после пары итерации. Фазовый сдвиг меняет знак фазы элементов в G, фазовый сдвиг меняет знак (начальное значение итерации) и оставляет другие элементы без изменений.

i. Ускорение через проверку

Цель: иногда нужно проверить правильность заявленного решения. Такие проверки могут затем использоваться для ускорения решения соответствующей проблемы. Часто бывает трудно найти решение проблемы, но проверить, действительно ли заявленное решение правильное, просто. Например, разложить число на множители сложно, но умножать числа просто. Проверка возможных решений осуществляется через оракул. Сканирование осуществляется средствами алгоритма Гровера, таким образом, $O(\sqrt{N})$ вызовов функции оракула определяет решение. Предпосылкой для этого шаблона является то, что решения могут быть обнаружены с помощью оракула.

j. Квантово-классическое взаимодействие

Цель: решение проблемы часто не достигается с помощью только квантового компьютера. Поэтому решение частично выполняется на классическом компьютере и частично на квантовом, и обе части решения взаимодействуют. Некоторые квантовые алгоритмы по своей сути требуют предварительной или постобработки на классическом оборудовании, в результате которого решение разбивается на классическую и квантовую части. Кроме того, если квантовый компьютер имеет небольшое количество кубитов или его вентили зашумлены, решение проблемы, возможно, придется разделить на часть, выполняемую на квантовом компьютере, и деталь, выполненную на классическом компьютере. *Решение:* важен сам факт того, что алгоритмы могут быть разделены. Как такое разделение применяется, зависит от конкретной проблемы.

Заключение

Конструирование алгоритмов для новых информационных технологий и специализированных вычислительных систем является динамичной областью, о чем свидетельствует количество существующих работ в данном направлении.

Разработками моделирующих систем, способных к поиску элементов (будь то число, массив чисел, набор символов или др. форма представления) в неструктурированной базе данных с использованием классических компьютеров, занимаются многие, и эта область давно не нова. Аналогичная же процедура в квантовых системах, при их разработке, потребует на один, а то и несколько порядков меньше операций, что делает разработку квантовых систем, нацеленных на решения данной проблемы, важной и полезной задачей.

***Благодарности*:** *работа выполнена при финансовой поддержке РФФИ, грант № 19-07-01082.*

***Acknowledgments*:** *The work was carried out with the financial support of RFFI, grant No. 19-07-01082.*

Список литературы:

1. Past, Present, Parallel: A Survey of Available Parallel Computer Systems / A. Trew, G. Wilson (ed.). Springer, 1991. 392 p.

2. Richard G. Milner. A Short History of Spin // Contribution to the XV International Workshop on Polarized Sources, Targets, and Polarimetry. Charlottesville, Virginia, USA, September 9–13, 2013. arXiv: 1311.5016. 16 p.

3. Гушанский С.М., Потапов В.С. Методика разработки и построения квантовых алгоритмов // Информатизация и связь. 2017. № 3. С. 101–104.

4. Kleppner D., Kolenkow R. An Introduction to Mechanics. 2nd Edishion. Cambridge: Cambridge University Press, 2014. 564 p.

5. Гушанский С.М., Поленов М.Ю., Потапов В.С. Реализация компьютерного моделирования системы с частицей в одномерном и двухмерном пространстве на квантовом уровне // Известия ЮФУ. Технические науки. Моделирование физических процессов и систем. Ростов-на-Дону: Изд-во ЮФУ, 2017. № 3. С. 223–233.

6. Hales L., Hallgren S. An improved quantum Fourier transform algorithm and applications: Proceedings of the 41st Annual Symposium on Foundations of Computer Science. 2000. November 12–14. P. 515.

7. Potapov V., Gushanskiy S., Polenov M. The Methodology of Implementation and Simulation of Quantum Algorithms and Processes // The 11th IEEE International Conference on Application of Information and Communication Technologies (AICT 2017). Institute of Electrical and Electronics Engineers. 2017. P. 437–441.

8. Attractive photons in a quantum nonlinear medium / Ofer Firstenberg, Mikhail D. Lukin [at al.] // Nature. 2013. Vol. 502, No. 7469. P. 71–75.

9. Нильсен М., Чанг И. Квантовые вычисления и квантовая информация = Quantum Computation and Quantum Information. Москва: Мир, 2006. 167 с.

10. Boneh D., Zhandry M. Quantum-secure message authentication codes // Proceedings of Eurocrypt. 2013. P. 592–608.

11. Chris Ferrie. Quantum Physics for Babies. Brdbk edition. Sourcebooks Jabberwocky, 2017. P. 23–24.

12. Wilde M., From Classical to Quantum Shannon Theory. 2011. arXiv: 1106.1445. DOI: 10.1017/9781316809976.001

13. Гузик В.Ф., Гушанский С.М., Потапов В.С. Количественные характеристики степени запутанности // Известия ЮФУ. Технические науки. 2016. № 3. С. 76–86.

14. Architecture and Software Implementation of a Quantum Computer Model / V. Potapov, S. Gushansky, V. Guzik, M. Polenov // Advances in Intelligent Systems and Computing. Springer Verlag, 2016. Vol. 465. P. 59–68.

15. Алгоритмы: построение и анализ = Introduction to Algorithms / Томас Х. Кормен, Ч.И. Лейзерсон, Р.Л. Ривест, К. Штайн. 2-е изд. Москва: Вильямс, 2006. 1296 с.

16. Cortese J.A., Braje T.M. Loading Classical Data into a Quantum Computer. 2018. arXiv: 1803.01958v1. 13 p.

17. Black-box quantum state preparation without arithmetic / Y.R. Sanders, G.H. Low, A. Scherer, D.W. Berry. Phys. 2018. arXiv: 1807.03206v1. DOI: 10.1103/PhysRevLett.122.020502

18. Kerenidis I., Prakash A. Quantum Recommendation Systems. 2016. arXiv: 1603.08675v3. 22 p.

19. Gilyén A., Arunachalam S., Wiebe N. Optimizing quantum optimization algorithms via faster quantum gradient computation. 2018. arXiv:1711.00465v3. DOI: 10.1137/1.9781611975482.87

20. Quantum computing with black-boxsubroutines / J. Thompson, M. Gu, K. Modi, V. Vedral. 2013. arXiv:1310.2927v5. DOI: 10.1088/1367-2630/aa99b3

References:

1. Past, Present, Parallel: A Survey of Available Parallel Computer Systems / A. Trew, G. Wilson (ed.). Springer, 1991. 392 p.

2. Richard G. Milner. A Short History of Spin // Contribution to the XV International Workshop on Polarized Sources, Targets, and Polarimetry. Charlottesville, Virginia, USA, September 9–13, 2013. arXiv: 1311.5016. 16 p.

3. Gushansky S.M., Potapov V.S. Methodology of development and construction of quantum algorithms // Informatization and Communication. 2017. No. 3. P. 101–104.

4. Kleppner D., Kolenkow R. An Introduction to Mechanics. 2nd Edishion. Cambridge: Cambridge University Press, 2014. 564 p.

5. Gushansky S.M., Polenov M.Yu., Potapov V.S. Implementation of computer modeling of a system with a particle in one-dimensional and two-dimensional space at the quantum level // News Bulletin of Southern Federal University. Technical Sciences. Modeling of Physical Processes and Systems. Rostov-on-Don: SFedU Publishing House, 2017. No. 3. P. 223–233.

6. Hales L., Hallgren S. An improved quantum Fourier transform algorithm and applications: Proceedings of the 41st Annual Symposium on Foundations of Computer Science. 2000. November 12–14. P. 515.

7. Potapov V., Gushanskiy S., Polenov M. The Methodology of Implementation and Simulation of Quantum Algorithms and Processes // The 11th IEEE International Conference on Application of Information and Communication Technologies (AICT 2017). Institute of Electrical and Electronics Engineers. 2017. P. 437–441.

8. Attractive photons in a quantum nonlinear medium / Ofer Firstenberg, Mikhail D. Lukin [at al.] // Nature. 2013. Vol. 502, No. 7469. P. 71–75.

9. Nielsen M., Chuang I. Quantum computation and quantum information. Moscow: Mir, 2006. 167 p.

10. Boneh D., Zhandry M. Quantum-secure message authentication codes // Proceedings of Eurocrypt. 2013. P. 592–608.

11. Chris Ferrie. Quantum Physics for Babies. Brdbk edition. Sourcebooks Jabberwocky, 2017. P. 23–24.

12. Wilde M., From Classical to Quantum Shannon Theory. 2011. arXiv: 1106.1445. DOI: 10.1017/9781316809976.001

13. Guzik V.F., Gushansky S.M., Potapov V.S. Quantitative characteristics of the degree of entanglement // News Bulletin of Southern Federal University. Technical Sciences. 2016. No. 3. P. 76–86.

14. Architecture and Software Implementation of a Quantum Computer Model / V. Potapov, S. Gushansky, V. Guzik, M. Polenov // Advances in Intelligent Systems and Computing. Springer Verlag, 2016. Vol. 465. P. 59–68.

15. Algorithms: construction and analysis = Introduc-tion to Algorithms / Thomas H. Cormen, Ch.I. Leiserson, R.L. Rivest, K. Stein. 2nd ed. Moscow: Williams, 2006.1296 p.

16. Cortese J.A., Braje T.M. Loading Classical Data into a Quantum Computer. 2018. arXiv: 1803.01958v1. 13 p.

17. Black-box quantum state preparation without arithmetic / Y.R. Sanders, G.H. Low, A. Scherer, D.W. Berry. Phys. 2018. arXiv: 1807.03206v1. DOI: 10.1103/PhysRevLett.122.020502

18. Kerenidis I., Prakash A. Quantum Recommendation Systems. 2016. arXiv: 1603.08675v3. 22 p.

19. Gilyén A., Arunachalam S., Wiebe N. Optimizing quantum optimization algorithms via faster quantum gradient computation. 2018. arXiv:1711.00465v3. DOI: 10.1137/1.9781611975482.87

20. Quantum computing with black-boxsubroutines / J. Thompson, M. Gu, K. Modi, V. Vedral. 2013. arXiv:1310.2927v5. DOI: 10.1088/1367-2630/aa99b3

Авторы заявляют об отсутствии конфликта интересов.

The authors declare no conflicts of interests.

Статья поступила в редакцию 02.04.2021; одобрена после рецензирования 04.05.2021; принята к публикации 08.05.2021.

The article was submitted 02.04.2021; approved after reviewing 04.05.2021; accepted for publication 08.05.2021.