УДК 512.6, 517.9, 519.6
DOI: 10.15827/0236-235X.140.618-630
Дата подачи статьи: 23.08.22, после доработки: 26.09.22
2022. Т. 35. № 4. С. 618–630

Оценка возможностей классических компьютеров при реализации симуляторов квантовых алгоритмов

П.В. Зрелов [1,2,3], *к.ф.-м.н., начальник отдела, zrelov@jinr.ru*
О.В. Иванцова [1,2], *научный сотрудник, ivancova@jinr.ru*
В.В. Кореньков [1,2,3], *д.т.н., директор лаборатории, korenkov@jinr.ru*
Н.В. Рябов [1,2], *инженер-программист, ryabov@jinr.ru*
С.В. Ульянов [1,2], *д.ф.-м.н., главный научный сотрудник, ulyanovsv46_46@mail.ru*

[1] *Объединенный институт ядерных исследований, Лаборатория информационных технологий им. М.Г. Мещерякова, г. Дубна, 141980, Россия*
[2] *Университет «Дубна», Институт системного анализа и управления, г. Дубна, 141980, Россия*
[3] *Российский экономический университет им. Г.В. Плеханова, г. Москва, 117997, Россия*

Современные квантовые устройства имеют серьезные ограничения на количество кубитов, ширину и глубину квантовой схемы. К тому же для них характерны сильные шумовые процессы, которые затрудняют получение корректного результата, заставляют проектировать квантовые схемы под конкретное квантовое устройство с учетом связи между кубитами, требуют квантовой коррекции ошибок. Применение классических компьютеров для симуляции квантовых вычислений позволяет избежать этих проблем. Они могут использоваться не только для быстрой проверки гипотез перед запуском на квантовых устройствах, но и для решения реальных задач.

В работе описаны проектирование и эффективное моделирование квантовых алгоритмов, подходы к разработке алгоритмов квантового поиска, алгоритм Гровера. С помощью квантовых симуляторов Qiskit и QuEST проведено исследование эффективности использования суперкомпьютера для симуляции квантовых схем на CPU и GPU на примере тестовой квантовой схемы и алгоритма Гровера. В статье дано описание алгоритма квантовой оценки фазы, являющейся базовым блоком в некоторых квантовых алгоритмах квантовой вычислительной физики и химии.

Симуляция алгоритма выполнена при помощи новейшего квантового симулятора cuQuantum от компании NVIDIA, который позволяет эффективно моделировать квантовые схемы на GPU с использованием множества графических процессоров, что существенно увеличивает скорость и позволяет выполнить алгоритм квантовой оценки фазы с достаточной точностью вычислений. В работе также отмечены сложности, с которыми можно столкнуться при симуляции различных алгоритмов с использованием большого количества кубитов или глубины схемы.

Ключевые слова: *квантовые вычисления, алгоритм Гровера, квантовая оценка фазы, квантовый симулятор, суперкомпьютер.*

Квантовое моделирование на классическом компьютере представляет особый интерес, поскольку вычисления на современных квантовых компьютерах имеют ограничения в виде небольшого числа кубитов, наличия случайных ошибок и отсутствия эффективного способа коррекции этих ошибок. Вследствие этих ограничений, по оценкам экспертов, использование универсальных квантовых компьютеров для решения практически значимых задач, вероятно, станет эффективным не ранее чем через 10 или даже 20 лет. Однако квантовые симуляторы на основе различных архитектур (атомных, молекулярных, оптических) могут быть полезны уже лет через пять [1]. И на данный момент, судя по всему, это направление является наиболее важным. Существуют реализации от Google, IBM, Rigetti, Intel, D-Wave (Google Research. Google Quantum Computing, IBM Research. IBM Quantum Computing). Одновременно развивается направление, связанное с программными квантовыми симуляторами для вычислений на компьютерах классической архитектуры с использованием CPU и GPU (Lisf of QC simulators).

В настоящей работе исследуется эффективность использования квантовых программных симуляторов для *квантовых алгоритмов* (КА). Для решения этой задачи были выбраны симуляторы QuEST и Qiskit, позволяющие исполь-

зовать как CPU (например, с применением библиотеки для параллельного программирования OpenMP), так и графические вычислители GPU. Эксперименты проводились на суперкомпьютере «Говорун» (http://hlit.jinr.ru/supercomputer_govorun/) в Лаборатории информационных технологий им. М.Г. Мещерякова Объединенного института ядерных исследований. Реализация КА на симуляторах с использованием классической архитектуры требует экспоненциального масштабирования памяти и времени вычислений [2].

Проведенные вычисления показали, что при моделировании на GPU можно достичь большего ускорения по сравнению с расчетами на CPU.

Следует отметить, что наибольшего ускорения расчетов, по утверждению компании NVIDIA, можно достичь с использованием нескольких графических процессоров [3, 4], что было продемонстрировано с помощью нового симулятора cuQuantum – сымитировано 3 375 кубитов на платформе NVIDIA DGX SuperPod™. Полноценные результаты экспериментов с использованием симулятора cuQuantum приводятся в отдельной работе.

Квантовые алгоритмы и их проектирование

Квантовые вычисления являются одним из наиболее бурно развивающихся направлений квантовых сквозных технологий. Квантовое моделирование обещает найти применение при решении целого ряда проблем физики конденсированных сред, физики высоких энергий, атомной физики, квантовой химии, космологии, машинного обучения, искусственного интеллекта [5].

Квантовые компьютеры, доступные в настоящее время, называют зашумленными квантовыми компьютерами промежуточного масштаба (NISQ) [6]. Они реализуют, как правило, от 50 до 100 кубитов, вследствие присутствия шумов при вычислениях возникают ошибки, а возможность их коррекции отсутствует.

Наряду с созданием квантовых компьютеров одним из важных является направление, связанное с разработкой алгоритмов для этих компьютеров, основанных на принципах, отличных от принципов классических алгоритмов.

Существуют различные научные подходы к формализации преобразований входных данных для вычисления выходных данных с использованием квантовых ресурсов. Модель квантовых вычислений определяется базовыми элементами, на которые декомпозируются вычисления.

Основной моделью, имеющей практическое значение, является гейтовая модель [7]. В этой модели вычисления разбиваются на последовательность из нескольких кубитных квантовых гейтов (вентилей). Это модель вычислений, заменяющая биты кубитами, а логические преобразования конечным набором унитарных гейтов, которые могут аппроксимировать любую произвольную унитарную операцию.

Алгоритм, выполненный в одной из моделей квантовых вычислений, называется КА. Разработка КА основана на принципах, отличных от принципов классических алгоритмов. Даже классические алгоритмы должны быть приведены в специальную (обратимую) форму, прежде чем их можно будет запустить на квантовом компьютере.

Визуализация типичного рабочего процесса КА на квантовом компьютере с гейтовой моделью показана на рисунке 1.

Особенностью этого процесса является конструирование алгоритма с помощью так называемых квантовых схем.

Выбор КА определяется характером задачи. На следующем шаге КА представляется в виде квантовой схемы как набор квантовых гейтов. Квантовый гейт (или квантовый вентиль) – это базовая квантовая схема, работающая на не-

Рис. 1. Визуализация типичного рабочего процесса КА на квантовом компьютере с гейтовой моделью

Fig. 1. Visualization of a typical workflow of a quantum algorithm on a quantum computer with a gate model

большом количестве кубитов. Гейты являются строительными блоками квантовых схем точно так же, как классические логические вентили для обычных цифровых схем, и подразделяются на однокубитовые и многокубитовые. Скомпилированная квантовая схема может быть либо выполнена на квантовом процессоре, либо реализована с помощью симулятора квантового компьютера.

Среди всего множества КА есть алгоритмы, имеющие явные преимущества перед известными классическими алгоритмами. Основной группой таких алгоритмов являются *алгоритмы квантового поиска* (АКП) [8–10]. Их особенностью является достаточность 75 % вероятности для выбранного решения (в отличие от классического метода достоверного поиска решения с вероятностью 1).

Структура КА

На рисунке 2 показана общая структура квантовой схемы.

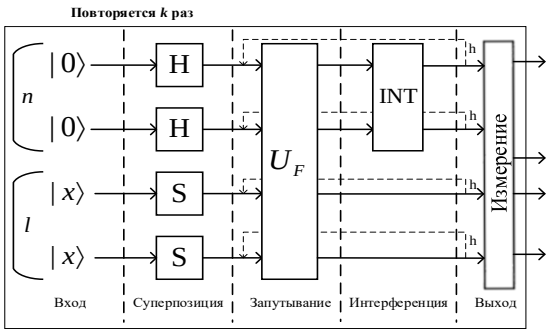


Рис. 2. Общая структура КА

Fig. 2. A quantum algorithm general structure

Процесс проектирования КА включает матричную форму трех квантовых операторов: суперпозиции, квантовой запутанности (или квантового оракула) и интерференции. Оператором суперпозиции является тензорное произведение *n* операторов Адамара и *l* операторов тождественного преобразования. Полученный таким образом оператор действует на первые *n* кубитов (первый регистр), создавая их суперпозицию, и действует тождественно на последние *l* кубитов (второй регистр), оставляя его без изменений. Вид оператора запутывания U_F зависит от свойств исходной функции *f*. Оператор интерференции зависит от рассматриваемого алгоритма. Оператором интерференции могут выступать, например, квантовое преобразование Фурье, оператор Адамара и др.

КА воздействует на исходный базисный вектор для генерации суперпозиции базисных векторов в качестве выходных данных. После создания суперпозиции выполняется измерение для извлечения информации об ответе. В квантовой механике измерение – это недетерминированная необратимая операция, которая выдает на выходе один из базисных векторов во входной суперпозиции. Вероятность того, что каждый базисный вектор будет результатом измерения, зависит от его амплитуды вероятности во входящей линейной комбинации.

Квантовые гейты и операция измерения составляют квантовый блок, который повторяется *k* раз, чтобы создать набор из *n* базисных векторов. Поскольку измерение является недетерминированной операцией, эти базисные векторы необязательно будут идентичными, и каждый базисный вектор кодирует часть информации, необходимой для решения задачи. Последняя часть алгоритма включает в себя интерпретацию собранных базисных векторов, чтобы получить окончательный ответ для исходной задачи с некоторой вероятностью.

На рисунке 3 показана типовая структура КА.

Сформировать разные модели квантовых вычислений можно, выбирая различные виды оператора U_F.

В общем виде модель квантовых вычислений состоит из следующих этапов:

– приготовление начального (классического или квантового) состояния $|\psi_{initial}\rangle$;

– преобразование Адамара (*H*) для начального состояния с целью формирования состояния суперпозиции;

Рис. 3. Типовая структура КА

Fig. 3. A quantum algorithm structure

– применение оператора запутанных состояний или оператора квантовой корреляции (квантового оракула) к состоянию суперпозиции;

– применение оператора интерференции;

– использование оператора измерения для результата квантовых вычислений $|\psi_{fin}\rangle$.

Процесс проектирования квантового алгоритма, как следует из рисунков 2 и 3, состоит из матричной формы представления трех операторов [11]: *Sup* – суперпозиции, U_F – квантовой корреляции (запутывания), или квантового оракула, и *Int* – интерференции.

В общем виде структуру квантового алгоритма, представленного на рисунке 3, можно описать следующим образом [11]:

$$C = [(Int \otimes^n I)\{U_F\}]^{h+1}[^n H \otimes^m S], \quad (1)$$

где I – тождественный оператор; символ $\otimes$ – тензорное произведение; H – оператор Адамара; S – I или H в зависимости от задачи. Основой квантового блока является квантовый гейт U_F, который зависит от свойств матрицы и применяется в алгоритме $h + 1$ раз. Первая часть процесса проектирования – выбор типа оператора в формуле (1), зависящего от проблемы запутывания. Оператор суперпозиции большинства АКП может быть выражен следующим образом: $Sup = \left(\bigotimes_{i=1}^{n} H\right) \otimes \left(\bigotimes_{i=1}^{m} S\right)$, где n и m – количество входов и выходов соответственно. Оператором может быть или оператор Адамара, или тождественный оператор в зависимости от алгоритма (табл. 1). В таблице QFT_n – квантовое преобразование Фурье; D_n – диффузия (инверсия относительно среднего).

Таблица 1

Параметры операторов суперпозиции и интерференции основных алгоритмов квантового поиска

Table 1

Parameters of superposition and interference operators of basic quantum search algorithms

Алгоритм	Суперпозиция	*m*	Интерференция
Гровера	$^nH \otimes H$	1	$D_n \otimes I$
Шора	$^nH \otimes {}^nI$	*n*	$QFT_n \otimes {}^nI$

Вычислительные модели алгоритмов квантового поиска

Основная сложность моделирования КА на классическом компьютере состоит в том, что ядром КА является унитарная матрица, размерность которой экспоненциально возрастает при линейном увеличении размерности квантовой системы.

На текущий момент существуют различные подходы к разработке АКП [12–14]. Авторы работы [15] выделяют среди них следующие.

1. Матричный подход, основанный на матричном представлении квантовых операторов. Этот подход является более стабильным и точным, но требует много памяти компьютера, что осложняет моделирование алгоритмов с большим числом кубитов.

2. Подход, основанный на алгоритмах, в которых элементы матрицы вычисляются по требованию. Этот подход не требует выделения памяти компьютера для квантовых операторов, поскольку вычисляет каждый компонент только тогда, когда он необходим, и позволяет использовать немного больший размер входных данных по сравнению с матричным подходом. К недостаткам можно отнести необходимость хранения в памяти компьютера вектора состояний и дополнительного изучения структуры операторов.

3. Проблемно-ориентированный подход. Количество переменных, используемых для представления переменной состояния в этом подходе, является постоянным. Фиксированное число переменных для представления вектора состояния позволяет пересмотреть традиционную схему моделирования квантового поиска. Операции матричного произведения заменяются арифметическими операциями с фиксированным числом параметров независимо от числа кубитов.

В данной статье рассматриваются квантовые симуляторы Qiskit и QuEST, которые реализуют подход, основанный на представлении квантовых матричных операторов.

Квантовый поиск Гровера. Задачу квантового поиска Гровера можно сформулировать следующим образом: из списка N предметов требуется определить один, удовлетворяющий какому-либо специфическому свойству.

На рисунке 4 приведена принципиальная схема алгоритма Гровера. Подробное описание алгоритма Гровера и его модификации приведены в [16, 17].

Выбор симуляторов. Среди большого количества квантовых программных симуляторов [18, 19] наиболее популярными являются следующие:

– платформа Qiskit для квантовых вычислений с открытым исходным кодом; разработана с учетом возможности расширения и под-

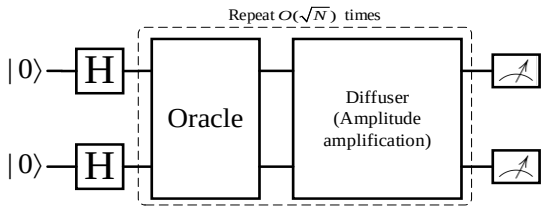


Рис. 4. Принципиальная схема алгоритма Гровера

Fig. 4. The Grover algorithm schematic diagram

держки исследований на квантовых системах IBM, основанных на сверхпроводящих кубитах, квантовых системах с ионной ловушкой IonQ и квантовых вычислительных устройствах AQT;

– конструктор квантовых схем Cirq, который можно использовать с квантовыми устройствами Google и с другими квантовыми симуляторами, в частности, с симулятором QSim;

– PyQuil, который можно использовать с квантовыми системами Rigetti;

– ProjectQ для поддержки исследований на квантовых системах IBM, AQT, AWS Braket, IonQ;

– cuQuantum – новейший симулятор от NVIDIA, который можно использовать вместе с Cirq и QSim или Qiskit (данную возможность планировали представить в версии 0.11.0, на момент публикации последняя версия 0.10.4).

Использование перечисленных квантовых устройств либо довольно дорого, либо ограничено небольшим количеством доступных кубитов.

В силу описанных ограничений запуск симуляции на классических компьютерах представляет особый интерес.

Симуляторы на классической архитектуре с использованием CPU и GPU

Из внушительного списка квантовых симуляторов на классической архитектуре были выбраны те, которые позволяют использовать OpenMP и GPU.

В настоящее время на CPU для квантовых симуляторов доступны возможности запуска на многопроцессорных системах, причем некоторые из симуляторов способны работать и в распределенной вычислительной среде. Это открывает возможности для симуляции систем с достаточно большим количеством кубитов.

Масштабируемость процесса симуляции на GPU развита меньше. Недавно NVIDIA представила собственный квантовый симулятор cuQuantum, который может работать на нескольких графических процессорах, связанных высокоскоростным интерфейсом NVLink. Один GPU с 32 Гб оперативной памяти может выполнить симуляцию 31 кубита. Вектор состояния для системы с бóльшим количеством кубитов уже не может поместиться в память одного GPU, и для дальнейшего масштабирования на каждый новый кубит требуется удваивать количество используемых GPU. В таком случае каждый GPU содержит часть вектора состояния и может применять гейты независимо. Если часть вектора состояния хранится на другом GPU, применение гейта к полному вектору может потребовать передачи значительного объема данных, измеряемого гигабайтами, а это возможно только при наличии быстрого соединения между GPU. Интерфейс NVLink позволяет производить очень быстрый обмен данными, что невозможно при использовании или нескольких GPU, подключенных к разным сокетам, или распределенных систем.

● QuEST – один из первых симуляторов, использующих многопоточность, распределенные вычисления и GPU-ускорители [20]. QuEST – это симулятор, представляющий чистые квантовые состояния с помощью векторов состояний и смешанные состояния с помощью матриц плотности. Количество памяти, необходимое для выполнения вычисления для вектора состояния, можно подсчитать по формуле $b \cdot 2^{\{qubits\text{-}29\}}$ (GiB), где $b = 8$ с двойной точностью. Следует отметить, что одновременное использование нескольких серверов или даже нескольких графических процессоров на одном сервере невозможно.

● Qiskit – на данный момент самый популярный квантовый симулятор. Он позволяет проводить моделирование на CPU и GPU, а недавно появилась возможность использовать OpenMP и CUDA [21]. Он также предоставляет возможность использовать различные методы моделирования. Каждый из них определяет внутреннее представление квантовой схемы и алгоритмы, используемые для обработки квантовых операций, имеет свои преимущества и недостатки, и выбор наилучшего метода является отдельным предметом исследования. В настоящем исследовании использован наиболее распространенный бэкэнд в Qiskit, который называется statevector_simulator [22], он возвращает квантовое состояние, которое представляет собой комплексный вектор размерности 2^n, где n – количество кубитов.

• QSim [23] содержит полный симулятор вектора состояния Шредингера и гибридный симулятор Шредингера–Фейнмана. Позволяет запускать симуляцию на процессоре с использованием OpenMP и GPU. На момент публикации cuQuantum от NVIDIA, имеющий большой потенциал для моделирования, интегрировался в QSim. Квантовый симулятор QSim поддерживается Google, в работе [24] приведены результаты его сравнения с квантовым устройством.

Аппаратное обеспечение

Как было упомянуто выше, эксперименты проводились на суперкомпьютере «Говорун» в МЛИТ ОИЯИ.

Эксперименты с графическим процессором осуществлялись на DGX-1 с 8X NVIDIA Tesla® V100 16 GB/GPU, 512 GB DDR4 RDIMM, 4X 20-Core Intel® Xeon® E5-2698 v4 2.2 GHz.

Моделирование с использованием центрального процессора проводилось на следующих конфигурациях:

– 8 серверов с 4X 72-Core Intel® Xeon Phi™ 7290 1.50 GHz, 96 GB DDR4;

– 4 сервера с 4X 24-Core Intel® Xeon® Platinum 8268 2.90 GHz, 192 GB DDR4.

Моделирование квантовых алгоритмов на компьютерах классической архитектуры. На этапе тестирования симуляторов QuEST и Qiskit были проведены эксперименты с использованием от 2 до 31 кубита, чтобы получить надлежащий масштаб для наблюдения за результатами. Для анализа результатов рассмотрим диапазон от 22 до 30 кубитов. Для меньшего количества кубитов симуляция занимает незначительное время. Симуляции для большего количества кубитов ограничены памятью.

При разработке схемы тестового алгоритма авторы воспользовались работой Google [25], произведя некоторые упрощения (рис. 5) и 100 повторений. Это значит, что к каждому кубиту в алгоритме применяется около 800 гейтов.

Схема сконструирована таким образом, чтобы можно было корректно сравнить два симулятора, потому что не все квантовые гейты для более сложных задач в рассматриваемых симуляторах реализованы одинаково. Например, алгоритм Гровера может быть реализован на различных симуляторах по-разному, что не позволяет произвести объективное сравнение.

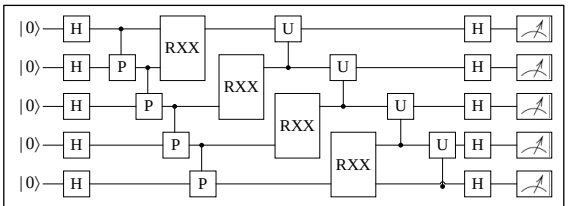


Рис. 5. Квантовая схема тестового алгоритма

Fig. 5. A quantum circuit of the test algorithm

На рисунке 6а представлены усредненные значения времени выполнения тестового алгоритма на обоих рассматриваемых симуляторах как на CPU, так и на GPU.

На обоих симуляторах на GPU по сравнению с CPU скорость выполнения алгоритма выше. Однако на QuEST нельзя произвести симуляцию более 29 кубит на GPU в силу ограничения памяти суперкомпьютера. Также стоит отметить, что Qiskit заметно быстрее QuEST на CPU со всей возможной параллелизацией.

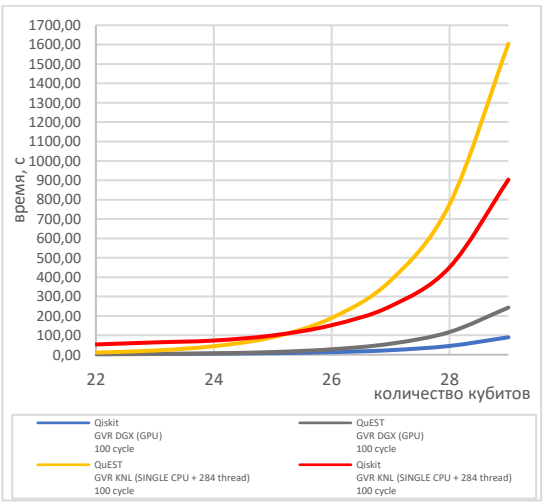
Алгоритм Гровера, изображенный на рисунке 4, также был реализован на обоих симуляторах (рис. 6б).

На рисунке 6б представлены усредненные значения времени выполнения алгоритма Гровера на обоих рассматриваемых симуляторах как на CPU, так и на GPU. Для 24 кубит – 1 shots, для 25 – 2 shots, для 26 – 4 shots и т.д., shots – количество запусков алгоритма.

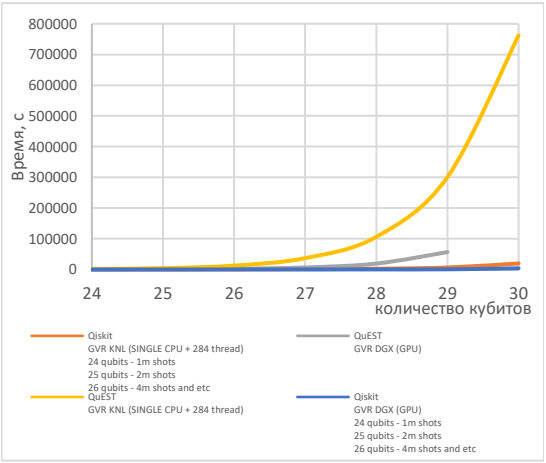
Основной результат – наименьшие временные затраты при выполнении данного алгоритма показывает Qiskit, а скорость выполнения на GPU меньше, чем на CPU.

Квантовая оценка фазы. Для понимания значимости квантового алгоритма оценки фазы и его основной идеи проведем краткий обзор основных методов, используемых для выполнения квантовой оценки фазы (Quantum Phase Estimation, QPE). QPE требует двух основных компонентов: использования метода моделирования с контролируемым гамильтонианом и иногда обратного квантового преобразования Фурье (QFT). В то время как исходный протокол квантовой фазовой оценки действительно использовал QFT, более современные версии, такие как байесовская квантовая фазовая оценка, избегают его применения. Этот последний подход также обладает свойством распараллеливания, реализации с минимальной классической постобработкой и требует меньшего количества кубитов.

Отметим особенности моделирования алгоритма QPE и его реализации.



а)



б)

Рис. 6. Время выполнения алгоритма на QuEST и Qiskit (CPU и GPU): а) тестового, б) Гровера

Fig. 6. Algorithm execution time on QuEST and Qiskit (CPU and GPU): a) test, б) Grover

Известно, что собственный вектор λ унитарной матрицы U удовлетворяет следующему уравнению: $U|\lambda\rangle = e^{2\pi i\theta}|\lambda\rangle$, где θ – угол или фаза, связанная с собственным значением матрицы U. Определение θ означает определение собственного значения $e^{2\pi i\theta}$. Квантовый алгоритм QPE предназначен для определения фазы θ и тем самым служит для определения собственного значения унитарного оператора. Предполагается, что $\theta \in [0, 2\pi)$ и представлена в виде двоичной дроби, для чего θ сначала приводится к интервалу [0, 1) с помощью $\theta^* = \theta/2\pi$, а затем записывается в виде $\theta^* = 0.\theta_1\theta_2\theta_3 \ldots \theta_n$, где n – точность, с которой эта двоичная дробь будет определена. Для простоты знак * у θ будет опускаться. Такая запись подразумевает, что $\theta^* = \theta_1 \cdot 2^{-1} + \theta_2 \cdot 2^{-2} + \theta_3 \cdot 2^{-3} + \ldots \theta_n \cdot 2^{-n}$.

Для построения алгоритма определения $\theta_1\theta_2\theta_3 \ldots \theta_n$ необходимо построить квантовую схему, в которой для n-битного двоичного представления используются n управляющих кубитов.

Квантовая схема для данного алгоритма состоит из двух наборов кубитов. Первый набор содержит измеряющие кубиты, которые используются для управления унитарными операциями, применяющимися ко второму набору. Измеряющие кубиты помещаются в суперпозицию с помощью гейта Адамара. Основная часть схемы заключается в применении унитарных операций ко второму набору кубитов. Эти унитарные операции являются управляемыми поворотами фаз на определенный угол, а угол – фазой, которую требуется оценить. Первый измеряющий кубит сделает 1 оборот, второй – 2 оборота, третий – 4 оборота и т.д. После этих поворотов к измеряющим кубитам применяется обратное QFT для преобразования из базиса Фурье в вычислительный базис. Затем происходит измерение кубитов.

В упрощенном виде схема с одним управляющим кубитом изображена на рисунке 7.

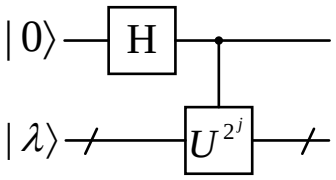


Рис. 7. Квантовая схема с одним управляющим кубитом

Fig. 7. A quantum circuit with one control qubit

На этой схеме управляющий кубит находится в суперпозиции состояний $|0\rangle$ и $|1\rangle$ вследствие действия оператора (гейта) Адамара (H), а $U^{2^j} = U \cdot U \ldots U$ (произведение унитарных операторов 2^j раз). Математически это может быть представлено следующим образом:

$$|0\rangle|\lambda\rangle \underline{H} \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle) \otimes |\lambda\rangle \underline{CU^{2^j}} \frac{1}{\sqrt{2}}(|0\rangle|\lambda\rangle + |1\rangle U^{2^j}|\lambda\rangle) = \frac{1}{\sqrt{2}}(|0\rangle|\lambda\rangle + |1\rangle e^{2\pi i(2^j\theta)}|\lambda\rangle),$$

где

$$2^j\theta = 2^j\left(\frac{\theta_1}{2^1} + \frac{\theta_2}{2^2} + \ldots \frac{\theta_n}{2^n}\right) = 2^{j-1}\theta_1 + 2^{j-2}\theta_2 + \ldots + \theta_j + \ldots + \frac{\theta_n}{2^{n-1}} = 0.\theta_1\theta_2\theta_3 \ldots \theta_j\theta_{j+1} \ldots \theta_n.$$

И тогда на выходе схемы получим

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.\theta_{j+1}\theta_{j+2}\ldots\theta_n)}|1\rangle)\otimes|\lambda\rangle\,. \qquad (2)$$

Например, пусть значение фазы θ равно 1/5, для ее определения с точностью до 4 бит необходимо применить матрицу U четыре раза (2^2), в результате будет получено 0011, что соответствует значению 0.1875 (вместо 0.2).

На рисунке 8 представлена схема с n управляющими кубитами. Для выходных кубитов справедливы выражения:

$$q_0 = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.\theta_1\theta_2\ldots\theta_n)}|1\rangle),$$

$$q_1 = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.\theta_2\theta_3\ldots\theta_n)}|1\rangle),$$

…

$$q_j = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.\theta_{j+1}\ldots\theta_n)}|1\rangle), \qquad (3)$$

…

$$q_{n-1} = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.\theta_n)}|1\rangle).$$

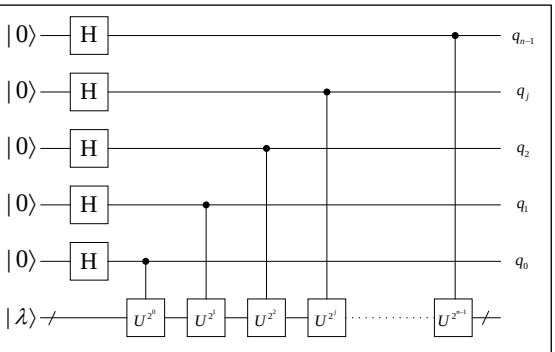


Рис. 8. Квантовая схема с n управляющими кубитами

Fig. 8. A quantum circuit with n control qubits

Чтобы получить представление фазы с точностью до n бит, необходимо биты $\theta_1\theta_2\theta_3 \ldots \theta_n$ как-то извлечь из кубитов q_0, q_1, …, q_{n-1}, представленных формулами (3). Это осуществляется с помощью обратного преобразования Фурье.

Если взять последний кубит q_{n-1} в (3), который содержит только один бит θ_n, то нетрудно проверить, что при $\theta_n = 0$ состояние, определяемое этим кубитом, будет $q_{n-1} = |+\rangle$, а при $\theta_n = 1$ $q_{n-1} = |-\rangle$. В X-базисе $|+\rangle$ соответствует $|0\rangle$, а $|+\rangle$ соответствует $|1\rangle$. Поэтому, если к последнему управляющему кубиту q_{n-1} применить гейт Адамара (H), при измерении кубита получим либо 0 с вероятностью 1, либо 1 с вероятностью 1 (рис. 9).

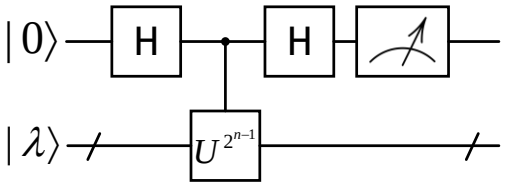


Рис. 9. Квантовая схема с измерением n-го управляющего кубита

Fig. 9. A quantum circuit with measuring the n-th control qubit

Но с другими кубитами не все так просто. В выражении для предпоследнего кубита q_{n-2} уже присутствуют 2 бита θ_{n-1} и θ_n, и второй из них явно лишний, и т.д. вплоть до первого кубита q_0, в котором присутствуют все биты: θ_1, θ_2, ..., θ_n. Рецепт избавления от лишних битов в управляемом вращающем гейте CROT. Допустим, нужно исключить бит θ_n, если $\theta_n = 0$, то гейт не применяется, если $\theta_n = 1$, то гейт поворачивает кубит так, чтобы исключить член $\theta_n/2^{n-1}$ из выражения для фазы и т.д. Все эти операции зашиты в блоке QFT^{-1} на схеме на рисунке 10, представляющем полную схему алгоритма квантовой оценки фазы.

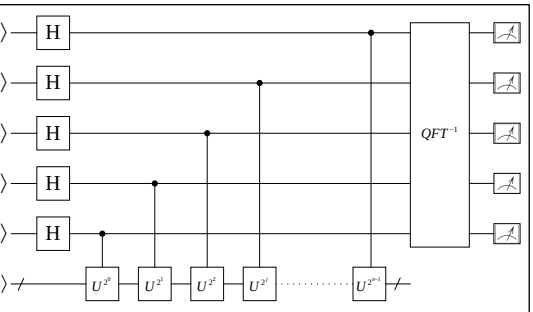


Рис. 10. Алгоритм квантовой оценки фазы

Fig. 10. A quantum phase estimation algorithm

Эксперименты проведены на задаче вычислений фазы для гейта с известным значением фазы, равным 1/3.

Алгоритм реализован с использованием конструктора квантовых схем Cirq и запущен на симуляторе Qsim с использованием симулятора cuQuantum на разном количестве кубитов. На рисунке 11 представлена квантовая схема решения такой задачи для трех измеряющих кубитов.

На рисунке 12а показан результат выполнения для квантовой схемы с тремя измеряющими кубитами. Наибольшую вероятность показал результат, равный 011, что в десятичной системе соответствует 3. При определении

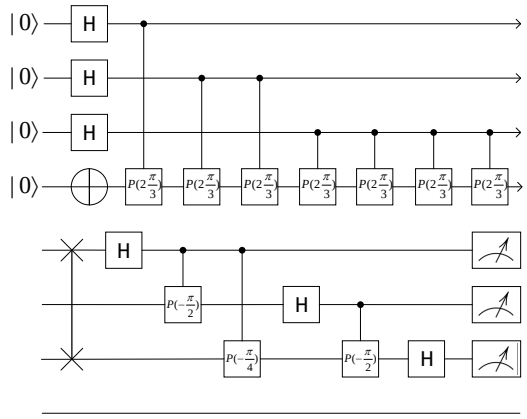
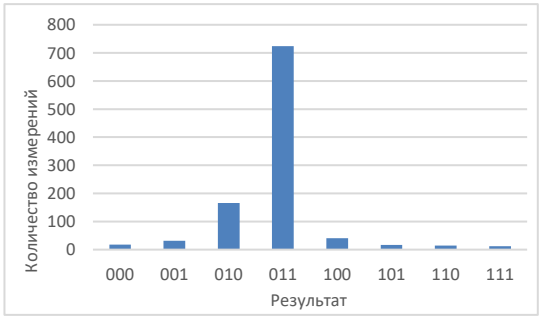
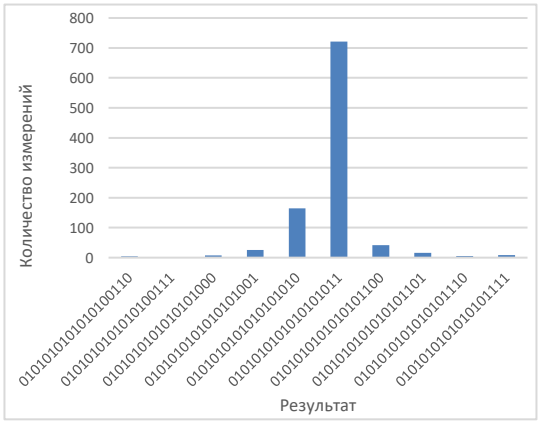


Рис. 11. Схема готового алгоритма для вычисления фазы на трех измеряющих кубитах

Fig. 11. A finished algorithmic diagram for phase calculation on 3 measuring qubits



а)



б)

Рис. 12. Результат выполнения QPE: а) для 3 кубитов, б) для 19 кубитов

Fig. 12. Quantum phase estimation: а) for 3 qubits, б) for 19 qubits

фазы, вычисляемой по формуле $3/2^N$, где N – количество кубитов, для случая $N = 3$ значение фазы равно 0.375. Увеличение количества измеряющих кубитов приводит к повышению точности результата: с каждым дополнительным кубитом добавляется еще один знак.

На рисунке 12б приведен результат для 19 кубитов (количество результатов сокращено для наглядности).

Наибольшую вероятность показал результат, равный 0101010101010101011, в десятичной системе это 174 763. Он позволяет получить значение фазы, равное 0.3333339691162109. Точность полученного при 19 кубитах решения существенно превосходит точность, полученную с меньшим количеством кубитов (например, 3, как в предыдущем примере).

Алгоритмы данного типа могут успешно применяться с использованием классических компьютеров уже сейчас, однако в отличие от ранее приведенного тестового алгоритма и алгоритма Гровера в рассматриваемом случае возникает совершенно другая проблема – возрастающая глубина схемы. Количество гейтов для выполнения операции U растет экспоненциально (табл. 2).

Таблица 2

Глубина квантовой схемы

Table 2

A quantum circuit depth

Количество кубитов	U	QFT	H	Глубина схемы
1	1	1	2	4
2	2	3	4	9
3	4	6	6	16
4	8	10	8	26
5	16	15	10	41
…				
16	32768	136	32	32936
17	65536	153	34	65723
18	131072	171	36	131279
19	262144	190	38	262372

Симуляция на классическом компьютере осложнена не только ограниченным количеством кубитов, но и ограниченной глубиной схемы. Осложняется не только сам процесс симуляции, но и процесс создания квантовой схемы, который выполняется на CPU. Этот процесс тоже сложен, потому что во многих симуляторах есть различные способы оптимизации, которые позволяют сэкономить немного ресурсов и памяти при вычислениях. Но есть и обратные ситуации, когда некоторые гейты реализованы как комбинация других гейтов, что неявно может увеличивать глубину схемы еще больше.

Заключение

Моделирование КА на симуляторах с использованием классической архитектуры в эпоху NISQ представляется перспективным еще на достаточно долгий период, вероятно, до момента появления квантовых процессоров с количеством кубитов более 10^5 и корректируемыми ошибками. До этого момента разработка и тестирование КА на симуляторах с использованием классической архитектуры, хотя и ограничены памятью компьютера или суперкомпьютера, позволили применять КА для решения ряда задач в самых различных областях: квантовой химии, развития нанотехнологий, сбора и обработки экспериментальных результатов для интеллектуального управления различными системами ускорителя в мегасайенс проекте НИКА, квантового глубокого машинного обучения в робототехнике и др.

Несмотря на имеющиеся ограничения на количество доступных кубитов, глубину схемы, скорость выполнения алгоритмов, квантовые симуляторы на компьютерах классической архитектуры позволяют осуществлять моделирование квантовых алгоритмов для решения задач в различных областях.

Рассмотренные в работе квантовые симуляторы QuEST и Qiskit дают существенный выигрыш в скорости при реализации всех рассмотренных алгоритмов при симуляции на графических процессорах по сравнению с реализацией на CPU. А моделирование на нескольких GPU на симуляторе cuQuantum дает еще больший выигрыш как в скорости, так и в количестве доступных кубитов.

Проведенный выбор квантовых симуляторов и их активное развитие позволяют приступить к формированию библиотеки КА для решения инженерных задач на суперкомпьютере «Говорун». Особое место в данной библиотеке отводится поисковому алгоритму Гровера и алгоритму оценки фазы.

В дальнейших планах продолжения данной работы – реализация быстрых модификаций алгоритма Гровера и других КА, формирующих ядро библиотеки для решения инженерных задач и задач в области квантовой химии.

Работа выполнена в рамках и при финансовой поддержке Минобрнауки РФ, грант № 075-10-2020-117.

Литература

1. Altman E., Brown K.R., Carleo G., Carr L.D. et al. Quantum simulators: Architectures and opportunities. PRX Quantum, 2021, vol. 2, no. 1, art. 017003. DOI: 10.1103/prxquantum.2.017003.

2. Alexeev Yu., Bacon D., Brown K.R., Calderbank R. et al. Quantum computer systems for scientific discovery. PRX Quantum, 2021, vol. 2, no. 1, art. 017001. DOI: 10.1103/PRXQuantum.2.017001.

3. Carrazza S., Cruz-Martinez J. VegasFlow: Accelerating Monte Carlo simulation across multiple hardware platforms. Computer Physics Communications, 2020, vol. 254, art. 107376. DOI: 10.1016/j.cpc.2020.107376.

4. LaRose R. Distributed memory techniques for classical simulation of quantum circuits. ArXiv, 2018, pp. 1–11. URL: https://arxiv.org/pdf/1801.01037.pdf (дата обращения: 17.07.2022).

5. Paudel H.P., Syamlal M., Crawford S.E., Lee Y.-L., Shugayev R.A. et al. Quantum computing and simulations for energy applications: Review and perspective. ACS Engineering Au, 2022, vol. 2, no. 3, pp. 151–196. DOI: 10.1021/acsengineeringau.1c00033.

6. Preskill J. Quantum computing in the NISQ era and beyond. Quantum, 2018, vol. 2, art. 79. DOI: 10.22331/q-2018-08-06-79.

7. Gyongyosi L., Imre S. Circuit depth reduction for gate-model quantum computers. Scientific Reports, 2020, vol. 10, no. 1, art. 11229. DOI: 10.1038/s41598-020-67014-5.

8. Shor P. Algorithms for quantum computation: discrete logarithms and factoring. Proc. XXXV Annual Symposium on Foundations of Computer Science, 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

9. Grover L.K. A fast quantum mechanical algorithm for database search. Proc. XXVIII Annual ACM STOC, 1996, pp. 212–219. DOI: 10.1145/237814.237866.

10. Giri P.R., Korepin V.E. A review on quantum search algorithms. Quantum Information Processing, 2017, vol. 16, art. 315. DOI: 10.1007/s11128-017-1768-7.

11. Ulyanov S.V. System and Method for Control Using Quantum Soft Computing. US Patent, no. US 6,578,018, B1, 2003.

12. Mehri-Dehnavi H., Dashtianeh H., Kuchaksaraei H.Y. et al. A Modified Quantum Search Algorithm. Int. J. of Theoretical Physics, 2018, vol. 57, pp. 3668–3681. DOI: 10.1007/s10773-018-3880-6.

13. Ulyanov S.V., Litvintseva L.V., Ulyanov S.S. Quantum Information and Quantum Computational Intelligence: Design & Classical Simulation of Quantum Algorithm Gates. Universita degli Studi di Milano: Polo Didattico e di Ricerca di Crema Publ., 2005, vol. 80.

14. Ulyanov S.V. et al. Quantum information and quantum computational intelligence: Classically efficient simulation of fast quantum algorithms (SW/HW Implementations). Note del Polo, Milan Univ., 2005, vol. 79.

15. Ivancova O.V., Korenkov V.V., Ulyanov S.V., Zrelov P.V. Quantum Software Engineering Toolkit. Quantum Fast Search Algorithms. Quantum Simulators on Classical Computers. Quantum Control Information Models. Part I. M.: KURS, 2022, 463 p.

16. Brickman K.A., Haljan P.C., Lee P.J., Acton M., Deslauriers L., Monroe C. Implementation of Grover's quantum search algorithm in a scalable system. Physical Review A, 2005, vol. 72, art. 050306. DOI: 10.1103/PhysRevA.72.050306.

17. Wang Y., Krstic P.S. Prospect of using Grover's search in the noisy-intermediate-scale quantum-computer era. Physical Review A, 2020, vol. 102, art. 042609.

18. Fingerhuth M., Babej T., Wittek P. Open source software in quantum computing. PLOS ONE, 2018, vol. 13, no. 12, art. e0208561. DOI: 10.1371/journal.pone.0208561.

19. Vidal G. Efficient classical simulation of slightly entangled quantum computations. Physical Review Letters, 2003, vol. 91, no. 14, art. 147902. DOI: 10.1103/physrevlett.91.147902.

20. Jones T., Brown A., Bush I., Benjamin S.C. QuEST and high performance simulation of quantum computers. Scientific Reports, 2019, vol. 9, no. 1, art. 10736. DOI: 10.1038/s41598-019-47174-9.

21. Qiskit. Running with Threadpool and DASK. URL: https://qiskit.org/documentation/apidoc/parallel.html (дата обращения: 17.07.2022).

22. Aleksandrowicz G., Alexander T., Barkoutsos P. et al. Qiskit: An open-source framework for quantum computing. Zenodo, 2021. URL: https://zenodo.org/record/2562111#.Y2EZZnZBzcs (дата обращения: 17.07.2022).

23. Quantum AI team and collaborators. Qsim. GitHub, 2020. URL: https://github.com/quantumlib/qsim (дата обращения: 17.07.2022).

24. Markov I. L., Fatima A., Isakov S.V., Boixo S. Quantum Supremacy is both closer and farther than it appears. ArXiv, 2018. URL: https://arxiv.org/abs/1807.10749 (дата обращения: 17.07.2022).

25. Arute F., Arya K., Babbush R., Bacon D. et al. Quantum supremacy using a programmable superconducting processor. Nature, 2019, vol. 574, pp. 505–510. DOI: 10.1038/s41586-019-1666-5.

Software & Systems
DOI: 10.15827/0236-235X.140.618-630

Received 23.08.22, Revised 26.09.22
2022, vol. 35, no. 4, pp. 618–630

Evaluating the capabilities of classical computers in implementing quantum algorithm simulators

P.V. Zrelov [1,2,3], *Ph.D. (Physics and Mathematics), Head of Department, zrelov@jinr.ru*
O.V. Ivantsova [1,2], *Research Associate, ivancova@jinr.ru*
V.V. Korenkov [1,2,3], *Dr.Sc. (Engineering), Director of the Laboratory, korenkov@jinr.ru*
N.V. Ryabov [1,2], *Engineer-Programmer, ryabov@jinr.ru*
S.V. Ulyanov [1,2], *Dr.Sc. (Physics and Mathematics), Chief Researcher, ulyanovsv46_46@mail.ru*

[1] *Joint Institute for Nuclear Research, Meshcheryakov Laboratory of Information Technologies, Dubna, 141980, Russian Federation*
[2] *Dubna State University, Institute of the System Analysis and Management, Dubna, 141980, Russian Federation*
[3] *Plekhanov Russian University of Economics, Moscow, 117997, Russian Federation*

Abstract. Modern quantum devices have severe limitations in the number of qubits, which limit the width and depth of the quantum circuit and have strong noise processes that make it difficult to obtain correct results. It is also necessary to design quantum circuits for a particular quantum device taking into account the coupling between qubits and to apply quantum error mitigation. It is possible to avoid these problems using classical computers to simulate quantum computation. Classical computers are used both for quick testing of hypotheses before running on quantum devices and for solving real-world problems.

The paper describes the process of designing and efficient modeling of quantum algorithms, approaches to developing quantum search algorithms, Grover's algorithm. Qiskit and QuEST quantum simulators were used to study the efficiency of using a supercomputer to simulate quantum circuits on CPUs and GPUs using the example of the quantum test circuit and Grover's algorithm. This paper describes a quantum phase estimation algorithm, which is a basic unit in some quantum algorithms of quantum computational physics and chemistry.

The algorithm is simulated using NVIDIA's newest cuQuantum quantum simulator. It allows efficient simulation of quantum circuits on GPUs using multiple GPUs, which significantly increases speed and allows the quantum phase estimation algorithm to be executed with sufficient computational accuracy. The paper also notes the difficulties when simulating different algorithms using a large number of qubits or circuit depth.

Keywords: quantum computing, Grover's algorithm, quantum phase estimation, simulation, quantum simulator, supercomputer.

Acknowledgements. *The work has been done and financially supported within the framework of the grant of the Ministry of Education and Science of the Russian Federation no. 075-10-2020-117.*

References

1. Altman E., Brown K.R., Carleo G., Carr L.D. et al. Quantum simulators: Architectures and opportunities. *PRX Quantum*, 2021, vol. 2, no. 1, art. 017003. DOI: 10.1103/prxquantum.2.017003.
2. Alexeev Yu., Bacon D., Brown K.R., Calderbank R. et al. Quantum computer systems for scientific discovery. *PRX Quantum*, 2021, vol. 2, no. 1, art. 017001, DOI: 10.1103/PRXQuantum.2.017001.
3. Carrazza S., Cruz-Martinez J. VegasFlow: Accelerating Monte Carlo simulation across multiple hardware platforms. *Computer Physics Communications*, 2020, vol. 254, art. 107376. DOI: 10.1016/j.cpc.2020.107376.
4. LaRose R. Distributed memory techniques for classical simulation of quantum circuits. *ArXiv*, 2018, pp. 1–11. Available at: https://arxiv.org/pdf/1801.01037.pdf (accessed 17.07.2022).
5. Paudel H.P., Syamlal M., Crawford S.E., Lee Y.-L., Shugayev R.A. et al. Quantum computing and simulations for energy applications: Review and perspective. *ACS Engineering Au*, 2022, vol. 2, no. 3, pp. 151–196. DOI: 10.1021/acsengineeringau.1c00033.
6. Preskill J. Quantum computing in the NISQ era and beyond. *Quantum*, 2018, vol. 2, art. 79. DOI: 10.22331/q-2018-08-06-79.
7. Gyongyosi L., Imre S. Circuit depth reduction for gate-model quantum computers. *Scientific Reports*, 2020, vol. 10, no. 1, art. 11229. DOI: 10.1038/s41598-020-67014-5.
8. Shor P. Algorithms for quantum computation: discrete logarithms and factoring. *Proc. XXXV Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.
9. Grover L.K. A fast quantum mechanical algorithm for database search. *Proc. XXVIII Annual ACM STOC*, 1996, pp. 212–219. DOI: 10.1145/237814.237866.
10. Giri P.R., Korepin V.E. A review on quantum search algorithms. *Quantum Information Processing*, 2017, vol. 16, art. 315. DOI: 10.1007/s11128-017-1768-7.
11. Ulyanov S.V. *System and Method for Control Using Quantum Soft Computing*. US Patent, no. US 6,578,018, B1, 2003.
12. Mehri-Dehnavi H., Dashtianeh H., Kuchaksaraei H.Y. et al. A Modified Quantum Search Algorithm. *Int. J. of Theoretical Physics*, 2018, vol. 57, pp. 3668–3681. DOI: 10.1007/s10773-018-3880-6.
13. Ulyanov S.V., Litvintseva L.V., Ulyanov S.S. *Quantum Information and Quantum Computational Intelligence: Design & Classical Simulation of Quantum Algorithm Gates*. Universita degli Studi di Milano: Polo Didattico e di Ricerca di Crema Publ., 2005, vol. 80.
14. Ulyanov S.V. et al. Quantum information and quantum computational intelligence: Classically efficient simulation of fast quantum algorithms (SW/HW Implementations). *Note del Polo, Milan Univ.*, 2005, vol. 79.
15. Ivantsova O.V., Korenkov V.V., Ulyanov S.V., Zrelov P.V. *Quantum Software Engineering Toolkit. Quantum Fast Search Algorithms. Quantum Simulators on Classical Computers. Quantum Control Information Models. Part I*. M.: KURS, 2022, 463 p.
16. Brickman K.A., Haljan P.C., Lee P.J., Acton M., Deslauriers L., Monroe C. Implementation of Grover's quantum search algorithm in a scalable system. *Physical Review A*, 2005, vol. 72, art. 050306. DOI: 10.1103/PhysRevA.72.050306.
17. Wang Y., Krstic P.S. Prospect of using Grover's search in the noisy-intermediate-scale quantum-computer era. *Physical Review A*, 2020, vol. 102, art. 042609.

18. Fingerhuth M., Babej T., Wittek P. Open source software in quantum computing. *PLOS ONE*, 2018, vol. 13, no. 12, art. e0208561. DOI: 10.1371/journal.pone.0208561.

19. Vidal G. Efficient classical simulation of slightly entangled quantum computations. *Physical Review Letters*, 2003, vol. 91, no. 14, art. 147902. DOI: 10.1103/physrevlett.91.147902.

20. Jones T., Brown A., Bush I., Benjamin S.C. QuEST and high performance simulation of quantum computers. *Scientific Reports*, 2019, vol. 9, no. 1, art. 10736. DOI: 10.1038/s41598-019-47174-9..

21. *Qiskit. Running with Threadpool and DASK*. Available at: https://qiskit.org/documentation/apidoc/parallel.html (accessed July 17, 2022).

22. Aleksandrowicz G., Alexander T., Barkoutsos P. et al. Qiskit: An open-source framework for quantum computing. *Zenodo*, 2021. Available at: https://zenodo.org/record/2562111#.Y2EZZnZBzcs (accessed July 17, 2022).

23. Quantum AI team and collaborators. Qsim. *GitHub*, 2020. Available at: https://github.com/quantumlib/qsim (accessed July 17, 2022).

24. Markov I. L., Fatima A., Isakov S.V., Boixo S. Quantum Supremacy is both closer and farther than it appears. *ArXiv*, 2018. Available at: https://arxiv.org/abs/1807.10749 (accessed July 17, 2022).

25. Arute F., Arya K., Babbush R., Bacon D. et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 2019, vol. 574, pp. 505–510. DOI: 10.1038/s41586-019-1666-5.

Для цитирования

Зрелов П.В., Иванцова О.В., Кореньков В.В., Рябов Н.В., Ульянов С.В. Оценка возможностей классических компьютеров при реализации симуляторов квантовых алгоритмов // Программные продукты и системы. 2022. Т. 35. № 4. С. 618–630. DOI: 10.15827/0236-235X.140.618-630.

For citation

Zrelov P.V., Ivantsova O.V., Korenkov V.V., Ryabov N.V., Ulyanov S.V. Evaluating the capabilities of classical computers in implementing quantum algorithm simulators. *Software & Systems*, 2022, vol. 35, no. 4, pp. 618–630 (in Russ.). DOI: 10.15827/0236-235X.140.618-630.