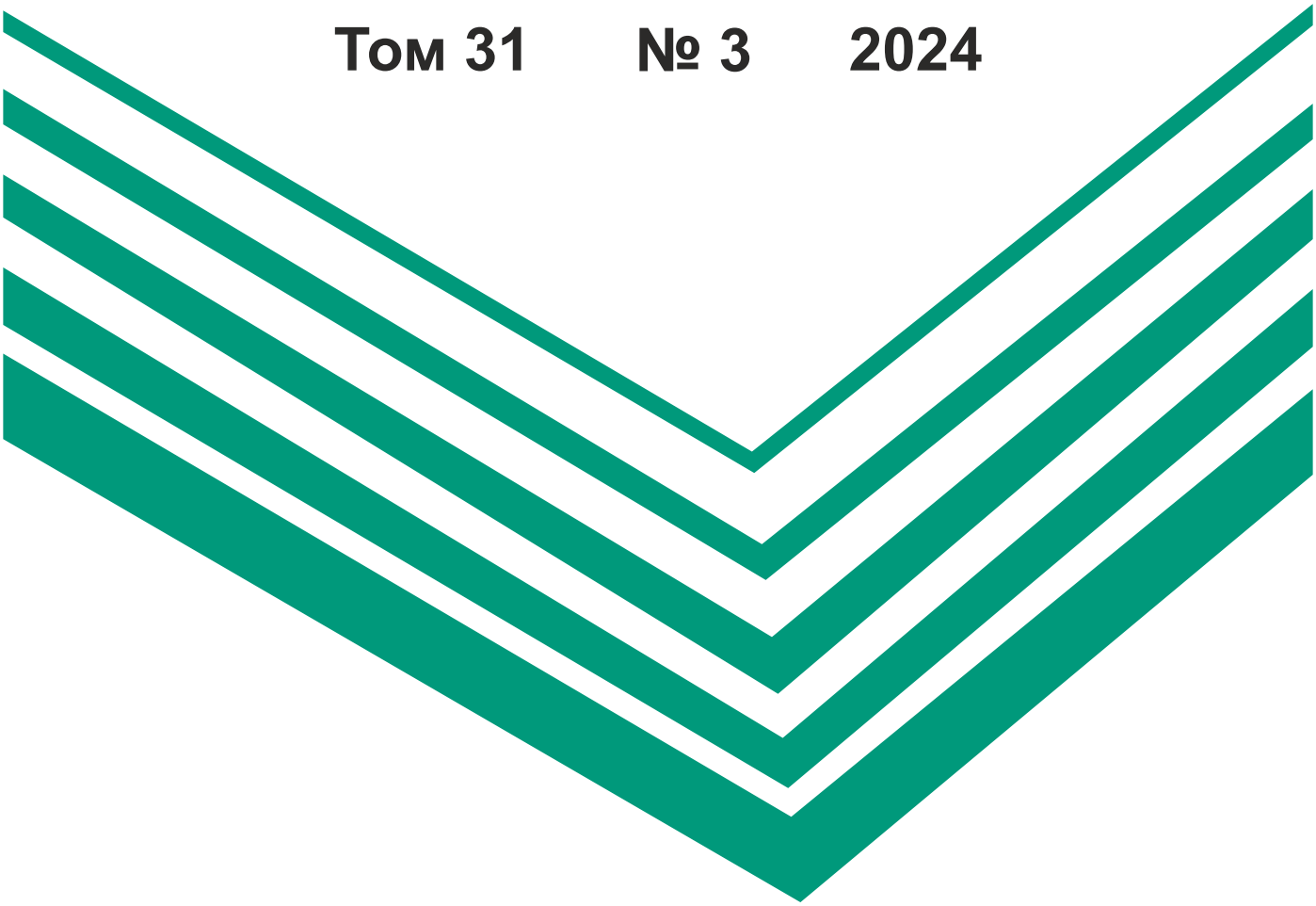
ISSN 2949-5598

ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ

Том 31 № 3 2024



Новосибирск
Издательство Института математики

ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Июль–сентябрь 2024. Т. 31, № 3. С. 24–53

УДК 519.7 DOI: 10.33048/daio.2024.31.777

НОВАЯ МОДЕЛЬ КВАНТОВОГО ОРАКУЛА ДЛЯ ГИБРИДНОЙ КВАНТОВО-КЛАССИЧЕСКОЙ АТАКИ НА ПОСТКВАНТОВЫЕ КРИПТОСИСТЕМЫ, ОСНОВАННЫЕ НА РЕШЁТКАХ

А. О. Бахарев

Новосибирский гос. университет,
ул. Пирогова, 2, 630090 Новосибирск, Россия

E-mail: a.bakharev@g.nsu.ru

Аннотация. Криптосистемы на основе решёток являются одними из основных постквантовых альтернатив асимметричной криптографии, используемой в настоящее время. Большинство атак на такие криптосистемы можно свести к задаче нахождения кратчайшего вектора в решётке (SVP). Ранее авторами была предложена модель квантового оракула из алгоритма Гровера для реализации гибридного квантово-классического алгоритма, основанного на алгоритме GaussSieve и решающего SVP. В настоящей работе предложена и проанализирована новая модель квантового оракула. Предложены и оценены две реализации новой модели квантового оракула. Проанализирована сложность реализации новой модели квантового оракула для атаки на постквантовые криптосистемы, основанные на решётках и являющиеся финалистами конкурса постквантовой криптографии NIST. Приведено сравнение полученных результатов для новой и уже существующей моделей квантового оракула. Табл. 4, ил. 10, библиогр. 48.

Ключевые слова: квантовый поиск, криптография с открытым ключом, криптография на решётках, постквантовая криптография, алгоритм Гровера, квантовые вычисления.

Введение

Развитие квантовых вычислений ведёт к необходимости в разработке и анализе криптосистем, устойчивых к атакам с использованием квантовых компьютеров — алгоритмов постквантовой криптографии [1–3]. Программа на квантовом компьютере может быть представлена квантовой

схемой. Ключевыми параметрами квантовых схем являются число кубитов, число вентилей и глубина схемы. Существует ряд открытых вопросов, связанных с квантовыми схемами, таких, как оценки сложности реализации квантовых схем, оптимальная реализация квантовых схем и др.

В 2016 г. Национальный институт стандартов и технологий США (NIST) опубликовал отчёт, в котором было проанализировано влияние квантовых вычислений на действующие стандарты шифрования. Согласно выводам отчёта симметричные криптосистемы (AES [4]) и хеш-функции (SHA-2, SHA-3 [5]) требуют увеличения размерностей ключей и входных последовательностей, а асимметричные криптосистемы (RSA [6], ECDSA, ECDH, DSA [7]) не являются постквантовыми. Большое влияние на отсутствие стойкости действующих стандартов асимметричного шифрования относительно квантовых вычислений приписывается алгоритму Шора [8], решающему задачи дискретного логарифмирования и факторизации за полиномиальное от длины двоичной записи порядка группы и факторизуемого числа время соответственно. Вследствие этого в том же году NIST объявил о начале конкурса «Post-Quantum Cryptography Competition», направленного на выявление новых — квантово устойчивых — стандартов цифровой подписи и инкапсуляции ключа. 5 июля 2022 г. завершился третий этап конкурса NIST [9], по итогам которого стандартами были выбраны криптосистемы, основанные на теории решёток и хеш-функциях. В феврале 2022 г. начался конкурс «Korean Post-Quantum Cryptography Competition», направленный на стандартизацию постквантовой криптографии в Южной Корее [10].

Настоящая работа посвящена криптоанализу криптосистем, основанных на решётках. Одной из задач в теории решёток является задача нахождения кратчайшего вектора (SVP), которая заключается в нахождении в заданной своим базисом решётке ненулевого вектора, имеющего наименьшую длину. В общем случае SVP является NP-трудной задачей [11]. Большинство атак на решётчатые криптосистемы сводится к нахождению вектора из решётки, отношение длины которого к длине вектора, являющегося решением SVP, не превосходит некоторого полинома от размерности решётки. Существует множество алгоритмов, решающих SVP, таких, как алгоритмы перечисления [12–14], алгоритмы редукции базиса [15], их комбинация [16–18], алгоритмы просеивания [19–23] и др. [24–26].

Известен ряд работ, направленных на разработку и анализ алгоритмов квантового криптоанализа, как симметричных [27–35], так и асимметричных криптосистем [36–39]. На данный момент остаётся открытым вопрос точных оценок параметров квантовых схем, используемых при квантовом криптоанализе. Для алгоритма GaussSieve в работе [40] уже

рассматривался данный вопрос, и была предложена модель квантового оракула, хранящая список векторов в квантовой памяти. В настоящей работе предлагается новая модель квантового оракула, хранящая список векторов в классической памяти. Под классической памятью подразумевается память, используемая обычным, не квантовым компьютером.

1. Основные определения и понятия

Пусть $\mathbb{F}_2$ — поле, состоящее из двух элементов, а $\mathbb{F}_2^n$ — векторное пространство размерности n над полем $\mathbb{F}_2$. *Весом* $wt(x)$ двоичного вектора $x \in \mathbb{F}_2^n$ называется число его ненулевых координат. Введём отношение частичного порядка $\preceq$ на множестве $\mathbb{F}_2^n$. Для $x, y \in \mathbb{F}_2^n$ положим

$$x \preceq y \Leftrightarrow x_i \leqslant y_i \quad \text{для любого } i \in \overline{1,n}.$$

Произвольная функция $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ называется *булевой функцией от n переменных*. Через $\oplus$ будем обозначать операцию сложения по модулю 2, т. е. $a \oplus b = (a + b) \bmod 2$. Любая булева функция f от n переменных единственным образом представляется в виде

$$f(x_1, \ldots, x_n) = \bigoplus_{k=1}^{n} \bigoplus_{i_1,\ldots,i_k} a_{i_1,\ldots,i_k} x_{i_1} \cdots x_{i_k} \oplus a_0,$$

где при каждом k все индексы $i_1, \ldots, i_k$ различны и $a_0, a_{i_1,\ldots,i_k} \in \mathbb{F}_2$. Такое представление называется *полиномом Жегалкина* или *алгебраической нормальной формой* функции f. *Степенью* булевой функции называется число переменных в самом длинном слагаемом АНФ булевой функции f. *Векторной булевой функцией* называется произвольное отображение $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$. Векторную булеву функцию $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ можно представить в виде $F(x) = (f_1(x), \ldots, f_m(x))$, где f_j — булева функция от n переменных. Функции f_j называют *координатными*.

1.1. Решётки. В настоящей работе рассматриваются криптосистемы, основанные на задачах из теории решёток.

Определение 1. Пусть $u_1, \ldots, u_d \in \mathbb{R}^n$, $d \leqslant n$, — линейно независимые векторы. *Решёткой* размерности d называется множество

$$\Lambda = \left\{ \sum_{i=1}^{d} b_i u_i \mid b_i \in \mathbb{Z} \right\}.$$

Линейно независимая система векторов, порождающая решётку, называется *базисом решётки*.

Пусть $p \geqslant 1$ — вещественное число. Тогда для вектора $x = (x_1, \ldots, x_n)$ из $\mathbb{R}^n$ *норма* l_p равна

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}.$$

Определение 2. *Задача нахождения кратчайшего вектора* (SVP) — найти в заданной своим базисом решётке ненулевой вектор, имеющий наименьшую длину относительно нормы l_p.

В данной работе рассматривается вариант SVP в евклидовой норме l_2. Далее для удобства вместо $\|\cdot\|_2$ будем писать $\|\cdot\|$.

В 2010 г. в [21] был предложен алгоритм GaussSieve, решающий SVP.

Алгоритм 1. Алгоритм GaussSieve (Миччанчо, Вулгарис, 2010)

Вход: B — базис решётки.
Выход: v — кратчайший вектор решётки.

1: Инициализировать пустой неупорядоченный список L и пустой стек S;
2: **repeat**
3: получить вектор v из стека (или сгенерировать новый);
4: **while** $w \leftarrow$ ПОИСК$\{w \in L \mid \|v \pm w\| \leqslant \|v\|\}$ **do**
5: уменьшить v с помощью w $(v \leftarrow v \pm w)$;
6: **while** $w \leftarrow$ ПОИСК$\{w \in L \mid \|w \pm v\| \leqslant \|w\|\}$ **do**
7: удалить w из списка L;
8: уменьшить w с помощью v $(w \leftarrow w \pm v)$;
9: добавить w в стек S;
10: **if** v изменился, **then** добавить v в стек S;
11: **else** добавить v в список L;
12: **until** v — кратчайший вектор;
13: **return** вектор v;

На вход алгоритма 1 поступает базис решётки, на основе которого будут строиться новые векторы при условии пустого стека S. Функция ПОИСК перебирает векторы w в списке и проверяет их на одно из условий поиска:

$$\|v \pm w\| \leqslant \|v\| \text{ или } \|w \pm v\| \leqslant \|w\|. \tag{1}$$

Если такой вектор существует, то функция возвращает его, иначе она прерывает первый цикл, в котором находится. Авторами [21] предложено эвристическое условие остановки, основанное на численных экспериментах. Таким образом, алгоритм работает до тех пор, пока не сработает эвристическое условие остановки.

Так как длина списка L увеличивается экспоненциально с ростом размерности решётки, самой трудозатратной операцией данного алгоритма является функция ПОИСК. В рамках предложенного в [38] подхода ускорение достигается за счёт использования в функции ПОИСК квантового алгоритма поиска.

1.2. Задача поиска. Задача, решаемая в функции ПОИСК, называется *задачей поиска*. Предполагается, что есть неупорядоченный список из K элементов, в котором, как минимум, один элемент удовлетворяет некоторому условию. Требуется найти по крайней мере один такой элемент. Другими словами, определена булева функция f, которая по номеру элемента (его двоичному представлению x) определяет, является ли элемент подходящим. Если элемент подходящий, то $f(x) = 1$, иначе $f(x) = 0$. В такой постановке задача поиска сводится к нахождению решения уравнения $f(x) = 1$.

В классическом варианте при условии, что решение одно, требуется $\sim K/2$ обращений к функции f для нахождения решения. Квантовый алгоритм поиска элемента в неупорядоченном списке (алгоритм Гровера [41]) решает данную задачу за $\sim \frac{\pi}{4}\sqrt{K}$ обращений к *оракулу* — квантовому аналогу функции f. О том, как булева функция моделируется на квантовом компьютере, будет рассказано далее.

2. Квантовые вычисления

Далее будут изложены необходимые сведения о квантовых вычислениях и принципах их работы. Более подробную информацию можно найти в работах [42, 43].

2.1. Кубит. Квантовый компьютер, в отличие от обычного, оперирует *квантовыми битами* (*кубитами*). Подобно классическому биту, который может находиться в состоянии 0 или 1, кубит имеет возможные состояния $|0\rangle$ и $|1\rangle$. Здесь используется *дираковское обозначение* $|\cdot\rangle$, которое является стандартным обозначением состояния в квантовой механике. Различие между битами и кубитами в том, что кубит может находиться в состоянии, отличном от $|0\rangle$ или $|1\rangle$. Можно составить *линейную комбинацию* состояний (*суперпозицию*):

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle.$$

Числа α и β комплексные, и $|\alpha|^2+|\beta|^2 = 1$. Иначе говоря, состояние одного кубита можно представить как единичный вектор из $\mathbb{C}^2$. Однако мы не можем измерить кубит, чтобы определить его квантовое состояние, т. е. значения α и β. Из квантовой механики следует, что при измерении кубита получаем либо результат $|0\rangle$ с вероятностью $|\alpha|^2$, либо результат $|1\rangle$ с вероятностью $|\beta|^2$.

Подобно случаю одиночного кубита система двух кубитов имеет четыре *состояния вычислительного базиса,* обозначаемых как $|00\rangle$, $|01\rangle$, $|10\rangle$ и $|11\rangle$. Здесь для любых $x, y \in \mathbb{F}_2$ выполнено $|xy\rangle \equiv |x\rangle|y\rangle$, где $|x\rangle$ — состояние первого кубита, а $|y\rangle$ — состояние второго кубита. Тогда вектор состояния, описывающий два кубита, имеет вид

$$|\varphi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle,$$

где $|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1$. Систему с произвольным числом кубитов описывает

Постулат 1. С каждой изолированной физической системой связывается комплексное векторное пространство со скалярным произведением, которое называется *пространством состояний* системы. Система полностью описывается *вектором состояния*, который представляет собой единичный вектор в пространстве состояний системы.

Для квантовых схем будем применять обозначения, представленные на рис. 1. В этих обозначениях временна́я ось направлена слева направо.

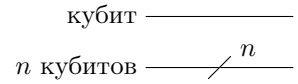


Рис. 1. Обозначения кубита и n кубитов

Постоянными будем называть кубиты, используемые на протяжении всей работы оракула, а *вре́менными* — те, которые используются только во время проведения промежуточных операций.

2.2. Эволюция квантовомеханической системы. Изменения состояния $|\psi\rangle$ квантовомеханической системы во времени описывает

Постулат 2. Эволюция замкнутой квантовой системы описывается унитарным преобразованием. Другими словами, состояние $|\psi\rangle$ системы в момент времени t_1 связано с её состоянием $|\psi'\rangle$ в момент времени t_2 посредством унитарного оператора U, зависящего только от моментов времени t_1 и t_2:

$$|\psi'\rangle = U|\psi\rangle. \tag{2}$$

Пусть $|\psi\rangle = |x_1, x_2, \ldots, x_k\rangle$ и $|\psi'\rangle = |y_1, y_2, \ldots, y_k\rangle$. Тогда равенство (2) можно переписать в обозначениях квантовых схем, как это показано на рис. 2. Базовое преобразование квантового компьютера будем называть *вентилем.* Для того чтобы описать работу вентиля, достаточно указать принцип работы данного вентиля на вычислительном базисе кубитов, на которых он действует. Для произвольного числа кубитов n вычислительный базис имеет вид $\{|x\rangle \mid x \in \mathbb{F}_2^n\}$. В настоящей работе для

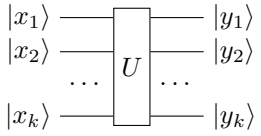
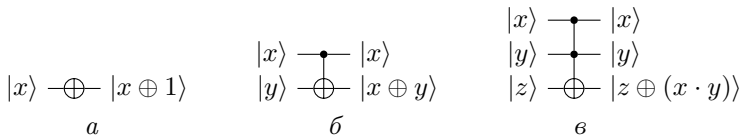


Рис. 2. Квантовая схема равенства (2)

построения всех операций и функций на квантовом компьютере используются базисные вентили, представленные на рис. 3 ($x, y, z \in \mathbb{F}_2$).



а *б* *в*

Рис. 3. Используемые вентили:
а) Паули-X (NOT), б) CNOT, в) Тоффоли (CCNOT)

Обозначение процесса измерения в вычислительном базисе кубита в квантовых схемах изображено на рис. 4. Здесь $\psi = 0$ с вероятностью $|\alpha|^2$ и $\psi = 1$ с вероятностью $|\beta|^2$.

Рис. 4. Измерение кубита

Пусть $x \in \mathbb{F}_2^n$, $y \in \mathbb{F}_2$ и U_f — квантовый аналог булевой функции f от n переменных. Тогда действие U_f на кубитах $|x\rangle$ и $|y\rangle$

$$U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$$

моделируется схемой на рис. 5.

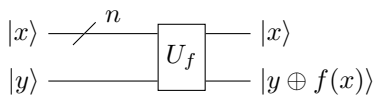


Рис. 5. Моделирование булевой функции на квантовом компьютере

Важным оператором, используемым во многих квантовых алгоритмах, является оператор Адамара H, изображённый на рис. 6. В матричном представлении этот оператор имеет вид

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

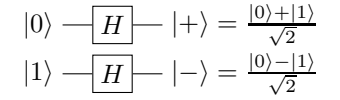


Рис. 6. Вентиль Адамара

Определим *глубину квантовой схемы* как число слоёв, которые содержит схема. Один слой состоит из базисных вентилей, применённых к непересекающимся множествам кубитов.

Так как информация, записанная во временных кубитах, не нужна после получения результата промежуточной операции, данные кубиты нужно очистить для возможности их дальнейшего использования. *Очистка кубитов* заключается в применении в обратном порядке ранее используемых вентилей. Значит, для очистки всех временных кубитов необходимо после получения результата операции применить в обратном порядке все ранее применённые вентили, которые не участвуют в изменении выходных кубитов операции.

2.3. Алгоритм Гровера. *Квантовый параллелизм* — это фундаментальное свойство многих квантовых компьютеров, позволяющее вычислять функцию $f(x)$ для многих различных значений x одновременно. Для понимания работы квантового параллелизма рассмотрим следующие рассуждения. Обозначим через $H^{\otimes n}$ преобразование Адамара, действующее на n кубитов. Результатом применения $H^{\otimes n}$ к кубитами, изначально находящимися в состоянии $|0\dots0\rangle$, будет состояние

$$H^{\otimes n}|0\dots0\rangle = \frac{1}{2^{n/2}} \sum_{x\in\mathbb{F}_2^n} |x\rangle.$$

Иными словами, преобразование Адамара приводит к суперпозиции всех состояний вычислительного базиса с одинаковыми коэффициентами. Тогда параллельное вычисление булевой функции $f(x)$ от n переменных может быть выполнено следующим образом. Приготавливаем $n+1$ кубитов в состоянии $|0\dots0\rangle$, затем применяем к первым n кубитам преобразование Адамара, после чего задействуем квантовую схему, реализующую U_f. Это даёт состояние

$$U_f\Big[\Big(\frac{1}{2^{n/2}} \sum_{x\in\mathbb{F}_2^n} |x\rangle\Big)|0\rangle\Big] = \frac{1}{2^{n/2}} \sum_{x\in\mathbb{F}_2^n} |x, f(x)\rangle. \tag{3}$$

Однако этим параллелизмом нельзя воспользоваться непосредственно. Измерение состояния (3) даёт значение $f(x)$ только для одного x. Для получения пользы от квантового параллелизма нужно иметь возможность «извлекать» информацию о более чем одном значении $f(x)$ из суперпозиции состояний (3).

Квантовым алгоритмом, решающим задачу поиска, является *алгоритм Гровера* [41] (рис. 7), в котором $1 \leqslant M \leqslant 2^{n-1}$ — число решений уравнения $f(x) = 1$, т. е.

$$M = \left|\left\{x \in \mathbb{F}_2^n \mid f(x) = 1\right\}\right|,$$

и G обозначает итерацию Гровера (рис. 8). Преобразование «Фаза» является известным вентилем, в отличие от вентиля «Оракул», который строится под каждую задачу отдельно.

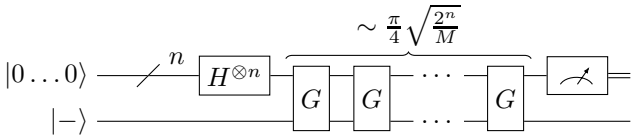


Рис. 7. Алгоритм Гровера

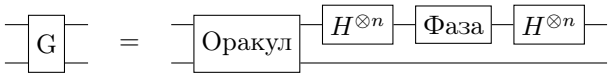


Рис. 8. Итерация Гровера

Принцип работы алгоритма Гровера можно найти в [40–43]. В [44] представлен подход, использующий алгоритм Гровера и позволяющий решать задачу поиска при неизвестном числе элементов, удовлетворяющих условию поиска.

3. Сложность реализации некоторых операций на квантовом компьютере

В [40] были получены реализации некоторых операций на квантовом компьютере, представленные в табл. 1, а также выражения для числа кубитов и глубины схемы, достаточных для реализации суммы нескольких целых положительных чисел.

Утверждение 1 [40]**.** *Пусть есть d целых положительных чисел, длина двоичного кода каждого из которых равна m. Тогда число кубитов, достаточное для их сложения, равно*

$$\sum_{i \in A} \left[\sum_{j=1}^{i-1} 2^{i-j-1}(m+j) + (m+i) \right] - (m + \min A),$$

где

$$A = \left\{ i \in \{1, 2, \ldots, \lceil \log_2(d+1) \rceil\} \mid \left\lfloor \frac{d}{2^{i-1}} \right\rfloor \bmod 2 \neq 0 \right\}.$$

Таблица 1

Число кубитов и глубина схемы, достаточные для реализации некоторых операций на квантовом компьютере [40]

Операция	Кубиты		Глубина
	постоянные	временные	
Сложение двух целых m-битных чисел, представленных в дополнительном коде	$m+1$	—	$2m+1$
Возведение в квадрат целого m-битного числа, представленного в прямом коде	$2m-2$	m^2-2m	$8m^2-24m+12$
Смена знака целого m-битного числа, представленного в дополнительном коде	m	—	$m+2$
Перевод целого m-битного числа из дополнительного кода в прямой	m	$\lceil\frac{m-1}{2}\rceil-1$	$2\lceil\log_2(m-1)+1\rceil+m$
Сравнение двух положительных целых m-битных чисел	1	$m+1$	$7m$

При этом глубина схемы равна

$$2\lceil\log_2 d\rceil\left(m+\frac{\lceil\log_2 d\rceil+1}{2}\right).$$

В табл. 1 и далее число постоянных и временных кубитов не учитывает входные кубиты схемы. Кроме этих операций для построения предложенной в разд. 4 модели квантового оракула понадобится также реализация векторной булевой функции.

3.1. Реализация векторной булевой функции, минимизирующая число кубитов. Рассмотрим реализацию произвольной векторной булевой функции $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ на квантовом компьютере. Обозначим через $|x_1\rangle,\ldots,|x_n\rangle$ кубиты, поступившие на вход функции F, а через $|y_1\rangle,\ldots,|y_m\rangle$ — кубиты на выходе функции F, инициализированные нулями.

Пусть булева функция $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ равна сумме всевозможных мономов от переменных $x_1,\ldots,x_n$. Описанный ниже метод реализации функции f даст оценку сверху на общее число вентилей и глубину схемы для реализации любой булевой функции от n переменных. Для временных

вычислений выделим дополнительно $n-2$ кубитов с начальной инициализацией $|0^{\otimes(n-2)}\rangle$ и обозначим их $|z_1\rangle, \ldots, |z_{n-2}\rangle$. Используем их для вычисления мономов. Первым шагом с помощью вентилей CCNOT последовательно присвоим в $|z_1\rangle$ состояние $|x_1 \cdot x_2\rangle$, $|z_2\rangle$ — состояние $|z_1 \cdot x_3\rangle$, $\ldots$, $|z_{n-2}\rangle$ — состояние $|z_{n-3} \cdot x_{n-1}\rangle$. После этого запишем с помощью вентиля CCNOT в $|y_1\rangle$ состояние $|z_{n-2} \cdot x_n\rangle = |x_1 \cdot \cdots \cdot x_n\rangle$, а с помощью вентиля CNOT — состояние, содержащееся в $|z_{n-2}\rangle$. Далее, используя вентили CCNOT, очистим кубит $|z_{n-2}\rangle$ и запишем в него $|z_{n-3} \cdot x_n\rangle$. Добавим в $|y_1\rangle$ состояние $|z_{n-2}\rangle$ и очистим $|z_{n-2}\rangle$. Таким образом можно добавить в $|y_1\rangle$ все возможные состояния, соответствующие мономам от переменных $x_1, \ldots, x_n$. Стоит отметить, что прибавление в $|y_1\rangle$ состояния, соответствующего моному первой степени, происходит за один шаг с помощью вентиля CNOT. Состояние $|1\rangle$ прибавляется в $|y_1\rangle$ с помощью действия вентиля NOT на $|y_1\rangle$. Тогда каждое состояние, соответствующее моному степени не равной 0, 1 и n, прибавляется в $|y_1\rangle$ за 2 вентиля CCNOT и один вентиль CNOT. Тогда общее число вентилей данной схемы равно $3 \cdot (2^n - n - 2) + n + 2$. Глубину схемы считаем равной общему числу вентилей.

Также процесс построения булевой функции, содержащей все мономы, можно представить как обход всех вершин некоторого полного двоичного дерева, начинающийся и заканчивающийся в корне. Вершины данного дерева являются мономами. Корень соответствует моному 1. Дети корневой вершины строятся следующим образом: если это вершина, соответствующая спуску влево, то её моном определяется как произведение монома, соответствующего родительской вершине, и переменной x_1; если это вершина, соответствующая спуску вправо, то её моном определяется как моном родительской вершины. Дети новых вершин определяются аналогичным образом с использованием переменной x_2. Данный процесс повторяется для всех переменных $x_1, \ldots, x_n$ до построения полного дерева. Так, самому левому листу данного дерева соответствует моном $x_1 \ldots x_n$, самому правому — моном 1. Представим дополнительно выделенные $n-2$ кубитов как стек. Тогда во время обхода каждый спуск влево, кроме спуска к вершинам с мономами степени 1 и n, соответствует добавлению в стек нового монома с помощью одного вентиля CCNOT. Подъём вправо соответствует удалению из стека монома с помощью одного вентиля CCNOT. Также каждый моном, полученный после спуска влево, необходимо добавить в $|y_1\rangle$, что соответствует использованию вентиля CNOT. Моном 1 добавляется в $|y_1\rangle$ с помощью вентиля NOT.

Рассмотрим случай, когда $m > 1$ и в каждой координатной булевой функции используются все возможные мономы. Этот случай будет являться оценкой сверху на общее число вентилей и глубину схемы. Тогда добавим в случай $m = 1$ некоторые изменения.

• После добавления в $|y_1\rangle$ состояния $|x_1 \cdot \dots \cdot x_n\rangle$, используя вентили CNOT, размножим данное состояние на все остальные $|y_i\rangle$, $i = 2, \dots, m$. Глубина такой операции составляет $\lceil \log_2 m \rceil$.

• Для состояний, соответствующих мономам степени, отличной от 0 и n, используем $\lceil \frac{m}{4} \rceil - 1$ дополнительных кубитов для клонирования состояния и параллельного добавления в $|y_1\rangle, \dots, |y_m\rangle$. После очищаем дополнительные кубиты. Глубина такой операции равна $2\lceil \log_2 m \rceil$.

Тогда глубина квантовой схемы функции $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ равна

$$(2\lceil \log_2 m \rceil + 2)(2^n - n - 2) + 2\lceil \log_2 m \rceil n + \lceil \log_2 m \rceil + 1 = \\ = 2^{n+1} - 2n + \lceil \log_2 m \rceil (2^{n+1} - 3) - 3.$$

3.2. Реализация векторной булевой функции, минимизирующая глубину схемы. Рассмотрим вторую реализацию векторной булевой функции. Аналогично первой реализации сначала покажем, как построить булеву функцию $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$. Заметим, что любую булеву функцию можно представить как $f = g_1 \oplus g_2 x_n$, где g_1, g_2 — булевы функции от переменных $x_1, \dots, x_{n-1}$. Аналогичное представление имеют функции $g_1 = h_{11} \oplus h_{12} x_{n-1}$ и $g_2 = h_{21} \oplus h_{22} x_{n-1}$, где $h_{11}, h_{12}, h_{21}, h_{22}$ — булевы функции от переменных $x_1, \dots, x_{n-2}$. Пусть построены функции $h_{11}, h_{12}, h_{21}, h_{22}$. Покажем, как с помощью вентилей CCNOT и входных кубитов $|x_{n-1}\rangle$ и $|x_n\rangle$ построить функцию f.

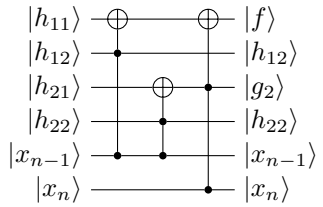


Рис. 9. Пример построения булевой функции

Тем самым каждую новую функцию предлагается разбивать на две функции от меньшего числа переменных до получения функций от переменной x_1. Таким образом, необходимо 2^{n-1} кубитов для всех булевых функций от переменной x_1. Так как булевы функции от переменных $x_1, \dots, x_k$ хранятся в кубитах, ранее используемых для хранения булевых функций от переменных $x_1, \dots, x_{k-1}$, для хранения всех функций, используемых в построении функции f, нужны 1 постоянный кубит для выхода функции f и $2^{n-1} - 1$ временных кубитов для промежуточных функций. Покажем, как построить функции от переменной x_1

(случай для получения верхней оценки). Предполагая, что каждая булева функция содержит моном x_1, добавим входное состояние, соответствующее данному моному, во все кубиты, выделенные для функций и инициализированные в начале значением $|0\rangle$. Эта операция описана в [45], и её глубина равна $\log_2(2^{n-1}) + 1$. Далее применим вентили NOT к тем кубитам, которые соответствуют функциям, содержащим константу 1. Далее построим функции от переменных x_1, x_2. Число таких функций равно 2^{n-1}. Для уменьшения глубины схемы предлагается размножить состояние $|x_2\rangle$ на 2^{n-1} временных кубитов так, как это описано в работе [45]. Глубина схемы, выполняющей данную операцию, равна $\log_2(2^{n-2})$. Затем, применяя вентили CCNOT к кубитам с состоянием $|x_2\rangle$ и кубитам, соответствующим функциям от переменной x_1, получим состояния, соответствующим функциям от переменных x_1, x_2. Повторим данный процесс до получения состояния, соответствующего функции f. Итоговая схема использует 1 постоянный кубит и

$$(2^{n-1} - 1) + (2^{n-2} - 1) + \cdots + (2^2 - 1) = \\ = \sum_{i=3}^{n} 2^{i-1} - (n-2) = \sum_{i=1}^{n} 2^{i-1} - (n+1) = 2^n - (n+2)$$

временных кубитов. Итоговая глубина схемы с очисткой временных кубитов равна

$$2(\log_2 2^{n-1} + 2 + \log_2 2^{n-2} + 1 + \cdots + \log_2 2^2 + 1 + 2) + 1 = n^2 + n + 1.$$

Теперь опишем построение векторной булевой функции F из $\mathbb{F}_2^n$ в $\mathbb{F}_2^m$. Сначала выделим m постоянных кубитов под выход операции. Так как векторную булеву функцию F можно представить как m булевых функций $f_1, \ldots, f_m \colon \mathbb{F}_2^n \to \mathbb{F}_2$, построение функции F состоит из размножения входного состояния до m состояний, параллельного вычисления состояний, соответствующих булевым функциям $f_1, \ldots, f_m$, и итоговой очистки

Таблица 2

Число кубитов и глубина схемы, достаточные для реализации функции $F \colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ на квантовом компьютере

Реализация	Минимизация кубитов	Минимизация глубины
Постоянные кубиты	m	m
Временные кубиты	$n + \left\lceil \frac{m}{4} \right\rceil - 3$	$(m-1)n + m(2^n - (n+2))$
Глубина	$2^{n+1} - 2n + \lceil \log_2 m \rceil (2^{n+1} - 3) - 3$	$2\log_2 m + n^2 + n + 1$

всей схемы. Тогда итоговое число временных кубитов равно

$$(m-1)n + m(2^n - (n+2)),$$

а итоговая глубина схемы равна

$$2\log_2 m + n^2 + n + 1.$$

3.3. Итоги. В табл. 2 представлены выражения для числа кубитов и глубины схемы, достаточных для предложенных реализаций векторной булевой функции $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ на квантовом компьютере.

4. Модель квантового оракула

В [40] была представлена модель квантового оракула, реализующая итерацию алгоритма Гровера для функции ПОИСК из алгоритма Gauss-Sieve. В этом разделе представлена новая модель данного оракула. Обозначим через K длину списка L, хранящего векторы размерности d, каждая координата которых кодируется битовой строкой длины m.

4.1. Описание модели. Рассмотрим модель оракула, представленную на рис. 10, у которой неупорядоченный список хранится в классической памяти.

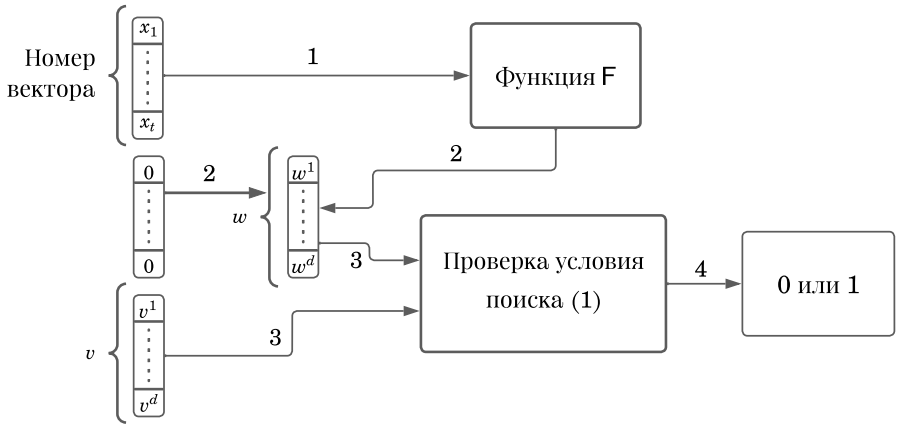


Рис. 10. Предлагаемая модель оракула с классическим списком

Работа данной модели оракула происходит следующим образом:

1) получение номера вектора на вход и передача его в функцию F;

2) получение в качестве выхода функции F вектора из списка L, соответствующего заданному номеру;

3) проверка полученного вектора на условие поиска (1);

4) вывод ответа: 1 — если вектор удовлетворяет условию, 0 — если нет.

Данная модель помогает избежать линейного роста числа кубитов, используемых оракулом, при увеличении размера списка L. Для этого построим векторную булеву функцию $\mathsf{F}\colon \mathbb{F}_2^{\lceil \log_2 K \rceil} \to \mathbb{F}_2^{dm}$, которая по номеру элемента списка возвращает соответствующий элемент. Если в списке нет соответствующего элемента, то F возвращает нулевой вектор. Поскольку в ходе алгоритма GaussSieve в списке отсутствует элемент, у которого каждая координата равняется нулю, возвращённый функцией F нулевой вектор будет указывать на отсутствие в списке соответствующего номеру элемента. Построение подобной векторной булевой функции может потребовать вычислений на классической части алгоритма, которые будут линейно зависеть от длины списка L. Однако не будем строить функцию F заново при каждом новом поиске, а будем использовать её перестроение, которое подробнее опишем ниже. Остальная часть оракула остаётся неизменной, поскольку построение булевой функции, соответствующей проверке на условие поиска, эквивалентно проверке каждого элемента списка на условие поиска.

Перестроение функции F. Чтобы избежать построения новой векторной булевой функции F при каждом изменении списка, будем использовать перестроение функции F. В ходе работы алгоритма GaussSieve между двумя поисками список может измениться следующим образом:

- из списка удалили один элемент;
- в список добавили один элемент.

В первом случае на место, в котором находился удалённый элемент, записывается нулевой вектор. Во втором случае возможны два варианта.

- В списке нет свободного места. Тогда необходимо расширить список, что потребует полного перестроения функции F.
- В списке есть свободное место. Тогда элемент занимает это свободное место.

Рассмотрим случаи, когда не требуется полного перестроения функции F. Пусть изменения произошли в элементе с номером k, тогда обозначим через $w_k^{\text{old}} \in \mathbb{F}_2^{dm}$ и $w_k^{\text{new}} \in \mathbb{F}_2^{dm}$ состояние этого элемента списка до и после изменения соответственно. Через $\mathsf{F}^{\text{old}}\colon \mathbb{F}_2^{\lceil \log_2 K \rceil} \to \mathbb{F}_2^{dm}$ обозначим функцию, соответствующую списку до изменения элемента с номером k, а через $\mathsf{F}^{\text{new}}\colon \mathbb{F}_2^{\lceil \log_2 K \rceil} \to \mathbb{F}_2^{dm}$ — функцию, соответствующую списку после изменения элемента с номером k. *Индикатором* неотрицательного целого числа $t \leqslant 2^{\lceil \log_2 K \rceil} - 1$ назовём булеву функцию $I_t\colon \mathbb{F}_2^{\lceil \log_2 K \rceil} \to \mathbb{F}_2$, определённую формулой

$$I_t(x) = \begin{cases} 1, & \text{если } x \text{ — двоичное представление } t, \\ 0 & \text{иначе.} \end{cases}$$

Пусть $\Delta_k = w_k^{\text{old}} \oplus w_k^{\text{new}}$. Тогда $\mathsf{F}^{\text{new}}(x)$ можно представить следующим образом:

$$\mathsf{F}^{\text{new}}(x) = \mathsf{F}^{\text{old}}(x) \oplus I_k(x) \cdot \Delta_k.$$

Такое преобразование может быть выполнено быстрее, чем построение новой векторной булевой функции.

4.2. Оценки сложности реализации, минимизирующей число кубитов квантовой схемы. В этом пункте рассматривается реализация предложенной модели квантового оракула, использующая реализацию функции F с меньшим числом кубитов.

Теорема 1. *Пусть имеется список длины K, состоящий из целочисленных векторов размерности $d \geqslant 2$, каждая координата которых кодируется битовой строкой длины $m \geqslant 3$. Тогда для реализации квантового оракула, представленного на рис.* 10, *потребуется не более*

$$\lceil \log_2 K \rceil + 14dm + 5d + 6m + 3\lceil \log_2 d \rceil + 3 + \\ + \max\left\{3d(m^2-1),\ \lceil \log_2 K \rceil + \left\lceil \frac{dm}{4} \right\rceil - 3\right\}$$

кубитов. При этом глубина не превосходит

$$2^{\lceil \log_2 K \rceil + 2} - 4\lceil \log_2 K \rceil + \lceil \log_2 dm \rceil (2^{\lceil \log_2 K \rceil + 2} - 6) + \\ + 16m^2 + 8m + 4\lceil \log_2 m \rceil + \lceil \log_2 d \rceil (2\lceil \log_2 d \rceil + 8m + 10) + 13.$$

Доказательство. Разделим доказательство на две части: оценка числа кубитов и оценка глубины схемы.

Оценка числа кубитов. Сначала оценим число постоянных кубитов. В отличие от представленной в работе [40] модели оракула, данная модель не использует переключатель и не хранит список L в квантовой памяти. Функция F использует dm постоянных кубитов, что соответствует dm кубитам, выделенным под копирование вектора в реализации [40, теорема 1], минимизирующей число кубитов. Поэтому, убирая из выражения для числа постоянных кубитов

$$\lceil \log_2 K \rceil + 2^{\lceil \log_2 K \rceil} + Kdm + 14dm + 5d + 6m + 3\lceil \log_2 d \rceil + 3$$

слагаемые Kdm и $2^{\lceil \log_2 K \rceil}$, соответствующие постоянным кубитам переключателя и кубитам, выделенным для хранения списка L, получаем число постоянных кубитов для предложенной реализации модели:

$$\lceil \log_2 K \rceil + 14dm + 5d + 6m + 3\lceil \log_2 d \rceil + 3.$$

Оценим число временных кубитов. Для реализации [40, теорема 1] это число равнялось временным кубитам $3d$ операций вычисления квадрата числа, представленного в прямом коде длины $m+1$. В данной же модели

вместо переключателя и копирования векторов используется функция F, а значит, число временных кубитов равно максимуму временных кубитов функции F и $3d$ операций вычисления квадрата числа, представленного в прямом коде длины $m+1$. Следовательно, общее число кубитов данной реализации модели квантового оракула равно

$$\lceil \log_2 K \rceil + 14dm + 5d + 6m + 3\lceil \log_2 d \rceil + 3 + \\ + \max\left\{3d(m^2-1),\ \lceil \log_2 K \rceil + \left\lceil \frac{dm}{4} \right\rceil - 3\right\}.$$

Оценка глубины схемы. В схеме из [40, теорема 1] глубина переключателя и копирования векторов из списка равнялась $(3^{\lceil \log_2 K \rceil} - 1)$ и $K(2\lceil \log_2 dm \rceil + 1)$ соответственно. Тогда, заменив в

$$2 \cdot 3^{\lceil \log_2 K \rceil} + 2K(2\lceil \log_2 dm \rceil + 1) + 16m^2 + 8m + \\ + 4\lceil \log_2 m \rceil + \lceil \log_2 d \rceil (2\lceil \log_2 d \rceil + 8m + 10) + 17$$

эти слагаемые на $2^{\lceil \log_2 K \rceil + 1} - 2\lceil \log_2 K \rceil + \lceil \log_2 dm \rceil (2^{\lceil \log_2 K \rceil + 1} - 3) - 3$ (глубину функции F), получим итоговую глубину схемы квантового оракула, в котором список хранится в классической памяти, с учётом очистки кубитов равную

$$2^{\lceil \log_2 K \rceil + 2} - 4\lceil \log_2 K \rceil + \lceil \log_2 dm \rceil (2^{\lceil \log_2 K \rceil + 2} - 6) + \\ + 16m^2 + 8m + 4\lceil \log_2 m \rceil + \lceil \log_2 d \rceil (2\lceil \log_2 d \rceil + 8m + 10) + 13.$$

Теорема 1 доказана.

Обозначим через $\mathcal{Q}$ схему, реализующую модель квантового оракула, представленного на рис. 10, и имеющую минимальное число кубитов. Тогда из теоремы 1 получим

Следствие 1. *В условиях теоремы* 1 *число кубитов, используемых в схеме $\mathcal{Q}$, не превосходит*

$$\lceil \log_2 K \rceil + 14dm + 5d + 6m + 3\lceil \log_2 d \rceil + 3 + \\ + \max\left\{3d(m^2-1),\ \lceil \log_2 K \rceil + \left\lceil \frac{dm}{4} \right\rceil - 3\right\}.$$

4.3. Оценки сложности реализации, минимизирующей глубину квантовой схемы. Здесь рассматривается реализация предложенной модели квантового оракула, использующая реализацию функции F с меньшей глубиной схемы.

Теорема 2. *Пусть имеется список длины K, состоящий из целочисленных векторов размерности $d \geqslant 2$, каждая координата которых кодируется битовой строкой длины $m \geqslant 3$. Тогда для реализации квантового*

оракула, представленного на рис. 10, *потребуется квантовая схема, глубина которой равна*

$$2\lceil\log_2 K\rceil^2 + 2\lceil\log_2 K\rceil + 4\lceil\log_2 dm\rceil + 16m^2 + 8m + \\ + 4\lceil\log_2 m\rceil + \lceil\log_2 d\rceil(2\lceil\log_2 d\rceil + 8m + 10) + 21.$$

При этом число кубитов данной схемы равно

$$\max\{3d(m^2-1),\ (dm-1)\lceil\log_2 K\rceil + dm(2^{\lceil\log_2 K\rceil} - (\lceil\log_2 K\rceil + 2))\} + \\ + \lceil\log_2 K\rceil + 14dm + 5d + 6m + 3\lceil\log_2 d\rceil + 3.$$

Доказательство. Аналогично теореме 1 разделим доказательство на две части: оценка числа кубитов и оценка глубины схемы.

Оценка числа кубитов. Так как данная реализация модели отличается от реализации модели из теоремы 1 только использованием реализации функции F, минимизирующей глубину схемы, число постоянных кубитов совпадает с числом постоянных кубитов из теоремы 1 и равно

$$\lceil\log_2 K\rceil + 14dm + 5d + 6m + 3\lceil\log_2 d\rceil + 3.$$

Теперь оценим число временных кубитов. Для первой реализации это число равнялось

$$\max\left\{3d(m^2-1),\ \lceil\log_2 K\rceil + \left\lceil\frac{dm}{4}\right\rceil - 3\right\}.$$

Тогда, заменив в этой формуле число кубитов для первого способа построения функции F, равное

$$\lceil\log_2 K\rceil + \left\lceil\frac{dm}{4}\right\rceil - 3,$$

числом кубитов для второго способа построения функции F, равным

$$(dm-1)\lceil\log_2 K\rceil + dm(2^{\lceil\log_2 K\rceil} - (\lceil\log_2 K\rceil + 2)),$$

получим число временных кубитов данной реализации

$$\max\{3d(m^2-1),\ (dm-1)\lceil\log_2 K\rceil + dm(2^{\lceil\log_2 K\rceil} - (\lceil\log_2 K\rceil + 2))\}.$$

Следовательно, общее число используемых кубитов равно

$$\max\{3d(m^2-1),\ (dm-1)\lceil\log_2 K\rceil + dm(2^{\lceil\log_2 K\rceil} - (\lceil\log_2 K\rceil + 2))\} + \\ + \lceil\log_2 K\rceil + 14dm + 5d + 6m + 3\lceil\log_2 d\rceil + 3.$$

Оценка глубины схемы. В схеме из теоремы 1 глубина реализации функции F вместе с очисткой равнялась

$$2^{\lceil\log_2 K\rceil+2} - 4\lceil\log_2 K\rceil + \lceil\log_2 dm\rceil(2^{\lceil\log_2 K\rceil+2} - 6) - 6.$$

Тогда, заменив в

$$2^{\lceil \log_2 K\rceil+2} - 4\lceil \log_2 K\rceil + \lceil \log_2 dm\rceil(2^{\lceil \log_2 K\rceil+2} - 6) + \\ + 16m^2 + 8m + 4\lceil \log_2 m\rceil + \lceil \log_2 d\rceil(2\lceil \log_2 d\rceil + 8m + 10) + 13.$$

эти слагаемые на

$$4\lceil \log_2(dm)\rceil + 2\lceil \log_2 K\rceil^2 + 2\lceil \log_2 K\rceil + 2$$

(глубину реализации функции F, минимизирующей глубину схемы, вместе с очисткой), получим итоговую глубину реализации предложенной модели квантового оракула с учётом очистки кубитов

$$2\lceil \log_2 K\rceil^2 + 2\lceil \log_2 K\rceil + 4\lceil \log_2 dm\rceil + 16m^2 + 8m + \\ + 4\lceil \log_2 m\rceil + \lceil \log_2 d\rceil(2\lceil \log_2 d\rceil + 8m + 10) + 21.$$

Теорема 2 доказана.

Обозначим через $\mathcal{D}$ схему, реализующую модель квантового оракула, представленного на рис. 10, и имеющую минимальную глубину. Тогда из теоремы 2 получим

Следствие 2. *В условиях теоремы* 2 *глубина схемы* $\mathcal{D}$ *не превосходит*

$$2\lceil \log_2 K\rceil^2 + 2\lceil \log_2 K\rceil + 4\lceil \log_2 dm\rceil + 16m^2 + 8m + \\ + 4\lceil \log_2 m\rceil + \lceil \log_2 d\rceil(2\lceil \log_2 d\rceil + 8m + 10) + 21.$$

4.4. Асимптотики. Важным параметром в полученных оценках является длина списка, которая на практике увеличивается экспоненциально с ростом размерности решётки d, т. е. $K \sim 2^{0{,}21d}$. Из теоремы 1 следует, что верхняя асимптотическая оценка для числа кубитов в реализации, минимизирующей число кубитов, второй модели оракула равна $O(\log_2 K + dm^2)$. Однако верхняя асимптотическая оценка для глубины данной реализации полиномиально зависит от длины списка K. Аналогично из теоремы 2 следует, что верхняя асимптотическая оценка для глубины реализации, минимизирующей глубину схемы, равна $O\big(\log_2^2 K + (m + \log_2 d)^2\big)$, но верхняя асимптотическая оценка для числа кубитов данной реализации полиномиально зависит от длины списка K.

5. Сравнение моделей

В этом разделе приведено сравнение новой модели квантового оракула с моделью из [40].

5.1. Связь числа кубитов и глубины схемы новой модели квантового оракула с параметрами постквантовых криптосистем. В табл. 3 для криптосистем NTRU [46], Saber [47], CRYSTALS-Kyber [48] указана связь уровней защищённости с числом кубитов и глубиной схемы, достаточных для новой модели квантового оракула.

Таблица 3

Число кубитов и глубина схемы, достаточные для предложенных реализаций новой модели квантового оракула

Реализация		Минимизация кубитов			Минимизация глубины		
Уровень защищённости		1	3	5	1	3	5
NTRU	кубиты	$2^{19,43}$	$2^{19,84}$	$2^{20,31}$	$2^{227,7}$	$2^{299,11}$	$2^{359,49}$
	глубина	$2^{219,91}$	2^{291}	2^{351}	$2^{16,92}$	$2^{17,64}$	$2^{18,15}$
SABER	кубиты	$2^{19,8}$	$2^{20,38}$	$2^{20,8}$	$2^{229,91}$	$2^{337,5}$	$2^{445,91}$
	глубина	$2^{221,91}$	2^{329}	2^{437}	2^{17}	2^{18}	$2^{18,75}$
CRYSTALS-Kyber	кубиты	$2^{19,62}$	$2^{20,21}$	$2^{20,62}$	$2^{229,81}$	$2^{337,4}$	$2^{445,81}$
	глубина	$2^{221,91}$	2^{329}	2^{437}	$2^{16,97}$	$2^{17,98}$	$2^{18,74}$

Сравнивая полученные численные результаты с результатами из [40], можно заметить, что хранение списка в классической памяти уменьшает число кубитов, используемых при атаках на криптосистемы. Также реализация предложенной в настоящей работе модели квантового оракула, минимизирующая число кубитов, имеет меньшую глубину схемы, чем аналогичная реализация предложенной ранее модели.

5.2. Сравнительный анализ моделей. В табл. 4 представлены асимптотические оценки сложности реализаций, минимизирующих число кубитов и минимизирующих глубины схемы, новой модели квантового оракула и модели из работы [40].

Как видно из таблицы, хранение списка в классической памяти позволяет избавиться от линейной зависимости от длины списка в асимптотических оценках числа кубитов. При этом асимптотические оценки глубины схемы либо остаются неизменными, либо некритично ухудшаются.

Заключение

Разработана и описана новая модель квантового оракула, применимая в алгоритме Гровера для реализации гибридно квантово-классического

Таблица 4

Асимптотические оценки сложности реализаций

Модель		Работа [40]	Настоящая работа
Минимизация кубитов	кубиты	$O(Kdm + dm^2)$	$O(\log_2 K + dm^2)$
	глубина	$O(K \log_2(dm) + (m + \log_2 d)^2)$	$O(K \log_2(dm) + (m + \log_2 d)^2)$
Минимизация глубины	кубиты	$O(K^2 + dm(K + m))$	$O(dm(K + m))$
	глубина	$O(\log_2 K + (m + \log_2 d)^2)$	$O\big(\log_2^2 K + (m + \log_2 d)^2\big)$

алгоритма на основе GaussSieve. Получены верхние оценки числа кубитов и глубины схемы на сложность реализации предлагаемой модели. Проведено сравнение новой модели с альтернативной моделью квантового оракула.

Финансирование работы

Исследование выполнено при поддержке Математического центра в Академгородке в рамках соглашения № 075–15–2022–282 с Министерством науки и высшего образования Российской Федерации. Дополнительных грантов на проведение или руководство этим исследованием получено не было.

Конфликт интересов

Автор заявляет, что у него нет конфликта интересов.

Литература

1. **Bernstein D. J.** Introduction to post-quantum cryptography // Post-quantum cryptography. Heidelberg: Springer, 2009. P. 1–14.
2. **Малыгина Е. С., Куценко А. В., Новосёлов С. А.** [и др.]. Постквантовые криптосистемы: открытые вопросы и существующие решения. Криптосистемы на решётках // Дискрет. анализ и исслед. операций. 2023. Т. 30, № 4. С. 46–90.
3. **Малыгина Е. С., Куценко А. В., Новосёлов С. А.** [и др.]. Постквантовые криптосистемы: открытые вопросы и существующие решения. Криптосистемы на изогениях и кодах, исправляющих ошибки // Дискрет. анализ и исслед. операций. 2024. Т. 31, № 1. С. 52–84.
4. **Daemen J., Rijmen V.** The design of Rijndael. Heidelberg: Springer, 2002. 238 p. DOI: 10.1007/978-3-662-04722-4.

5. **Dworkin M. J.** SHA-3 standard: Permutation-based hash and extendable-output functions. Gaithersburg, MD: Nat. Inst. Stand. Technol., 2015. 37 p. (Fed. Inf. Process. Stand. Publ.; V. 202). DOI: 10.6028/NIST.FIPS.202.
6. **Rivest R. L., Shamir A., Adleman L.** A method for obtaining digital signatures and public-key cryptosystems // Commun. ACM. 1978. V. 21, No. 2. P. 120–126.
7. **Barker E.** Digital signature standard (DSS). Gaithersburg, MD: Nat. Inst. Stand. Technol., 2013. 131 p. (Fed. Inf. Process. Stand. Publ.; V. 186-4). DOI: 10.6028/NIST.FIPS.186-4.
8. **Shor P. W.** Algorithms for quantum computation: Discrete logarithms and factoring // Proc. 35th Annu. Symp. Foundations of Computer Science (Santa Fe, USA, Nov. 20–22, 1994). Los Alamitos, CA: IEEE Comput. Soc., 1994. P. 124–134.
9. **Alagic G., Apon D., Cooper D.** [et al.]. Status report on the third round of the NIST post-quantum cryptography standardization process. Nat. Inst. Stand. Technol. Interagency Internal Rep. NIST IR 8413-upd1. Gaithersburg, MD: NIST, 2022. 102 p. DOI: 10.6028/NIST.IR.8413-upd1.
10. Korean Post-Quantum Cryptography Competition. Seoul: Natl. Intell. Serv., 2023. URL: kpqc.or.kr/competition.html (accessed: 9.10.2023).
11. **Micciancio D.** Inapproximability of the shortest vector problem: Toward a deterministic reduction // Theory Comput. 2012. V. 8, No. 1. P. 487–512.
12. **Kannan R.** Improved algorithms for integer programming and related lattice problems // Proc. 15th Annu. ACM Symp. Theory of Computing (Boston, USA, Apr. 25–27, 1983). New York: ACM, 1983. P. 193–206.
13. **Fincke U., Pohst M.** Improved methods for calculating vectors of short length in a lattice, including a complexity analysis // Math. Comput. 1985. V. 44, No. 170. P. 463–471.
14. **Gama N., Nguyen P. Q., Regev O.** Lattice enumeration using extreme pruning // Advances in cryptology — EUROCRYPT 2010. Proc. 29th Annu. Int. Conf. Theory and Application of Cryptographic Techniques (French Riviera, May 30 – June 3, 2010). Heidelberg: Springer, 2010. P. 257–278. (Lect. Notes Comput. Sci.; V. 6110).
15. **Lenstra A. K., Lenstra H. W., Lovász L.** Factoring polynomials with rational coefficients // Math. Ann. 1982. V. 261, No. 4. P. 515–534. DOI: 10.1007/BF01457454.
16. **Chen Y., Nguyen P. Q.** BKZ 2.0: Better lattice security estimates // Advances in cryptology — ASIACRYPT 2011. Proc. 17th Int. Conf. Theory and Application of Cryptology and Information Security (Seoul, South Korea, Dec. 4–8, 2011). Heidelberg: Springer, 2011. P. 1–20. (Lect. Notes Comput. Sci.; V. 7073).
17. **Schnorr C. P.** A hierarchy of polynomial time lattice basis reduction algorithms // Theor. Comput. Sci. 1987. V. 53, No. 2–3. P. 201–224. DOI: 10.1016/0304-3975(87)90064-8.
18. **Schnorr C. P., Euchner M.** Lattice basis reduction: Improved practical algorithms and solving subset sum problems // Math. Program. 1994. V. 66. P. 181–199. DOI: 10.1007/BF01581144.

19. **Becker A., Ducas L., Gama G., Laarhoven T.** New directions in nearest neighbor searching with applications to lattice sieving // Proc. 27th Annu. ACM-SIAM Symp. Discrete Algorithms (Arlington, VA, USA, Jan. 10–12, 2016). Philadelphia, PA: SIAM, 2016. P. 10–24.
20. **Herold G., Kirshanova E., Laarhoven T.** Speed-ups and time–memory trade-offs for tuple lattice sieving // Public-key cryptography — PKC 2018. Proc. 21st IACR Int. Conf. Practice and Theory of Public-Key Cryptography (Rio de Janeiro, Brazil, Mar. 25–29, 2018). Pt. I. Cham: Springer, 2018. P. 407–436. (Lect. Notes Comput. Sci.; V. 10769).
21. **Micciancio D., Voulgaris P.** Faster exponential time algorithms for the shortest vector problem // Proc. 21st Annu. ACM-SIAM Symp. Discrete Algorithms (Austin, TX, USA, Jan. 17–19, 2010). Philadelphia, PA: SIAM, 2010. P. 1468–1480.
22. **Nguyen P. Q., Vidick T.** Sieve algorithms for the shortest vector problem are practical // J. Math. Cryptol. 2008. V. 2, No. 2. P. 181–207. DOI: 10.1515/JMC.2008.009.
23. **Pujol X., Stehlé D.** Solving the shortest lattice vector problem in time $2^{2.465n}$. San Diego: Univ. California, 2009. (Cryptol. ePrint Archive; ID 2009/605). URL: eprint.iacr.org/2009/605 (accessed: 9.10.2023).
24. **Aggarwal D., Dadush D., Regev O., Stephens-Davidowitz N.** Solving the shortest vector problem in 2^n time using discrete Gaussian sampling // Proc. 47th ACM Symp. Theory of Computing (Portland, OR, USA, June 14–17, 2015). New York: ACM, 2015. P. 733–742.
25. **Doulgerakis E., Laarhoven T., de Weger B.** Finding closest lattice vectors using approximate Voronoi cells // Post-quantum cryptography. Rev. Sel. Pap. 10th Int. Conf. (Chongqing, China, May 8–10, 2019). Cham: Springer, 2019. P. 3–22. (Lect. Notes Comput. Sci.; V. 11505).
26. **Micciancio D., Voulgaris P.** A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations // Proc. 42nd ACM Symp. Theory of Computing (Cambridge, MA, USA, June 5–8, 2010). New York: ACM, 2010. P. 351–358.
27. **Денисенко Д. В., Никитенкова М. В.** Применение квантового алгоритма Гровера в задаче поиска ключа блочного шифра SDES // Журн. эксперим. и теор. физики. 2019. Т. 155, вып. 1. С. 32–53.
28. **Денисенко Д. В, Маршалко Г. Б., Никитенкова М. В., Рудской В. И., Шишкин В. А.** Оценка сложности реализации алгоритма Гровера для перебора ключей алгоритмов блочного шифрования ГОСТ Р 34.12-2015 // Журн. эксперим. и теор. физики. 2019. Т. 155, вып. 4. С. 645–653.
29. **Almazrooie M., Samsudin A., Abdullah R., Mutter K. N.** Quantum exhaustive key search with simplified-DES as a case study // SpringerPlus. 2016. V. 5, No. 1. P. 1–19.
30. **Dong X., Dong B., Wang X.** Quantum attacks on some Feistel block ciphers // Des. Codes Cryptogr. 2020. V. 88, No. 6. P. 1179–1203. DOI: 10.1007/s10623-020-00741-y.

31. **Frixons P., Naya-Plasencia M., Schrottenloher A.** Quantum boomerang attacks and some applications // Selected areas in cryptography. Proc. 28th Int. Conf. (Virtual Event, Sept. 29 – Oct. 1, 2021). Cham: Springer, 2021. P. 332–352. (Lect. Notes Comput. Sci.; V. 13203).
32. **Jaques S., Naehrig M., Roetteler M., Virdia F.** Implementing Grover oracles for quantum key search on AES and LowMC // Advances in cryptology — EUROCRYPT 2020. Proc. 39th Annu. Int. Conf. Theory and Application of Cryptographic Techniques (Zagreb, Croatia, May 10–14, 2020). Pt. II. Cham: Springer, 2020. P. 280–310. (Lect. Notes Comput. Sci.; V. 12106).
33. **Grassl M., Langenberg B., Roetteler M., Steinwandt R.** Applying Grover's algorithm to AES: quantum resource estimates // Post-quantum cryptography. Proc. 7th Int. Workshop (Fukuoka, Japan, Feb. 24–26, 2016). Cham: Springer, 2016. P. 29–43. (Lect. Notes Comput. Sci.; V. 9606).
34. **Langenberg B., Pham H., Steinwandt R.** Reducing the cost of implementing the advanced encryption standard as a quantum circuit // IEEE Trans. Quantum Eng. 2020. V. 1. P. 1–12.
35. **Zou J., Wei Z., Sun S., Liu X., Wu W.** Quantum circuit implementations of AES with fewer qubits // Advances in cryptology — ASIACRYPT 2020. Proc. 26th Int. Conf. Theory and Application of Cryptology and Information Security (Daejeon, South Korea, Dec. 7–11, 2020). Pt. II. Cham: Springer, 2020. P. 697–726. (Lect. Notes Comput. Sci.; V. 12492).
36. **Albrecht M. R., Gheorghiu V., Postlethwaite E. W., Schanck J. M.** Estimating quantum speedups for lattice sieves // Advances in cryptology — ASIACRYPT 2020. Proc. 26th Int. Conf. Theory and Application of Cryptology and Information Security (Daejeon, South Korea, Dec. 7–11, 2020). Pt. II. Cham: Springer, 2020. P. 583–613. (Lect. Notes Comput. Sci.; V. 12492).
37. **Gidney C., Ekerå M.** How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits // Quantum. 2021. V. 5. P. 433. DOI: 10.22331/q-2021-04-15-433.
38. **Laarhoven T., Mosca M., van de Pol J.** Finding shortest lattice vectors faster using quantum search // Des. Codes Cryptogr. 2015. V. 77, No. 2–3. P. 375–400.
39. **Perriello S., Barenghi A., Pelosi G.** A complete quantum circuit to solve the information set decoding problem // Proc. 2021 IEEE Int. Conf. Quantum Computing and Engineering (Broomfield, CO, USA, Oct. 17–22, 2021). Los Alamitos, CA: IEEE Comput. Soc., 2021. P. 366–377.
40. **Бахарев А. О.** Оценки сложности реализации квантового криптоанализа постквантовых криптосистем, основанных на решётках // Дискрет. анализ и исслед. операций. 2023. Т. 30, № 3. С. 5–42.
41. **Grover L. K.** A fast quantum mechanical algorithm for database search // Proc. 28th ACM Symp. Theory of Computing (Philadelphia, PA, USA, May 22–24, 1996). New York: ACM, 1996. P. 212–219. DOI: 10.1145/237814.237866.
42. **Nielsen M. A., Chuang I. L.** Quantum computation and quantum information. Cambridge: Camb. Univ. Press, 2010. 676 p.

43. Китаев А. Ю., Шень А. Х., Вялый М. Н. Классические и квантовые вычисления. М.: МЦНМО; ЧеРо, 1999. 192 с.
44. Boyer M., Brassard G., Høyer P., Tapp A. Tight bounds on quantum searching // Fortschr. Phys. 1998. V. 46, No. 4–5. P. 493–505.
45. Moore C., Nilsson M. Parallel quantum computation and quantum codes // SIAM J. Comput. 2001. V. 31, No. 3. P. 799–815.
46. Chen C., Danba O., Hoffstein J. [et al.]. NTRU algorithm specifications and supporting documentation. Eindhoven: Eindh. Univ. Technol., 2019. URL: `https://ntru.org/f/ntru-20190330.pdf` (accessed: 9.10.2023).
47. D'Anvers J.-P., Karmakar A., Roy S. S., Vercauteren F. SABER: Mod-LWR based KEM. Leuven: KU Leuven, 2017. URL: `https://www.esat.kuleuven.be/cosic/pqcrypto/saber/files/saberspecround1.pdf` (accessed: 9.10.2023).
48. Avanzi R., Bos J., Ducas L. [et al.]. CRYSTALS-Kyber algorithm specifications and supporting documentation. Amsterdam: Cent. Wiskd. Inform., 2021. URL: `https://cryptojedi.org/papers/kybernist-20171130.pdf` (accessed: 9.10.2023).

Бахарев Александр Олегович

Статья поступила
27 июня 2023 г.
После доработки —
27 ноября 2023 г.
Принята к публикации
22 марта 2024 г.

DISKRETNYI ANALIZ I ISSLEDOVANIE OPERATSII
/DISCRETE ANALYSIS AND OPERATIONS RESEARCH/
July–September 2024. Vol. 31, No. 3. P. 24–53

UDC 519.7 DOI: 10.33048/daio.2024.31.777

A NEW QUANTUM ORACLE MODEL FOR A HYBRID QUANTUM-CLASSICAL ATTACK ON POST-QUANTUM LATTICE-BASED CRYPTOSYSTEMS

A. O. Bakharev

Novosibirsk State University,
2 Pirogov Street, 630090 Novosibirsk, Russia
E-mail: a.bakharev@g.nsu.ru

Abstract. Lattice-based cryptosystems are one of the main post-quantum alternatives to asymmetric cryptography currently in use. Most attacks on these cryptosystems can be reduced to the shortest vector problem (SVP) in a lattice. Previously, the authors proposed a quantum oracle model from Grover's algorithm to implement a hybrid quantum-classical algorithm based on the GaussSieve algorithm and solving SVP. In this paper, a new model of a quantum oracle is proposed and analyzed. Two implementations of the new quantum oracle model are proposed and estimated. The complexity of implementing the new quantum oracle model to attack post-quantum lattice-based cryptosystems that are finalists of the NIST post-quantum cryptography competition is analyzed. Comparison of obtained results for new and existing models of quantum oracle is given. Tab. 4, illustr. 10, bibliogr. 48.

Keywords: quantum search, public-key cryptography, lattice-based cryptography, post-quantum cryptography, Grover's algorithm, quantum computation.

References

1. **D. J. Bernstein,** Introduction to post-quantum cryptography, in *Post-Quantum Cryptography* (Springer, Heidelberg, 2009), pp. 1–14.
2. **E. S. Malygina, A. V. Kutsenko, S. A. Novoselov,** [et al.], Post-quantum cryptosystems: Open problems and solutions. Lattice-based cryptosystems, *Diskretn. Anal. Issled. Oper.* **30** (4), 46–90 (2023) [Russian] [*J. Appl. Ind. Math.* **17** (4), 767–790 (2023)].

English transl.: *Journal of Applied and Industrial Mathematics* **18** (3), 395–411 (2024), DOI: 10.1134/S1990478924030037.

3. **E. S. Malygina, A. V. Kutsenko, S. A. Novoselov,** [et al.], Post-quantum cryptosystems: Open problems and solutions. Isogeny-based and code-based cryptosystems, *Diskretn. Anal. Issled. Oper.* **31** (1), 52–84 (2024) [Russian] [*J. Appl. Ind. Math.* **18** (1), 103–121 (2024)].
4. **J. Daemen** and **V. Rijmen,** *The Design of Rijndael* (Springer, Heidelberg, 2002), DOI: 10.1007/978-3-662-04722-4.
5. **M. J. Dworkin,** SHA-3 standard: Permutation-based hash and extendable-output functions (Nat. Inst. Stand. Technol., Gaithersburg, MD, 2015) (Fed. Inf. Process. Stand. Publ., Vol. 202), DOI: 10.6028/NIST.FIPS.202.
6. **R. L. Rivest, A. Shamir,** and **L. Adleman,** A method for obtaining digital signatures and public-key cryptosystems, *Commun. ACM* **21** (2), 120–126 (1978).
7. **E. Barker,** Digital signature standard (DSS) (Nat. Inst. Stand. Technol., Gaithersburg, MD, 2013) (Fed. Inf. Process. Stand. Publ., Vol. 186-4), DOI: 10.6028/NIST.FIPS.186-4.
8. **P. W. Shor,** Algorithms for quantum computation: Discrete logarithms and factoring, in *Proc. 35th Annu. Symp. Foundations of Computer Science, Santa Fe, USA, Nov. 20–22, 1994* (IEEE Comput. Soc., Los Alamitos, CA, 1994), pp. 124–134.
9. **G. Alagic, D. Apon, D. Cooper,** [et al.], Status report on the third round of the NIST post-quantum cryptography standardization process, *Nat. Inst. Stand. Technol. Interagency Internal Rep. NIST IR 8413-upd1* (NIST, Gaithersburg, MD, 2022), DOI: 10.6028/NIST.IR.8413-upd1.
10. Korean Post-Quantum Cryptography Competition (Natl. Intell. Serv., Seoul, 2023), URL: kpqc.or.kr/competition.html (accessed: 9.10.2023).
11. **D. Micciancio,** Inapproximability of the shortest vector problem: Toward a deterministic reduction, *Theory Comput.* **8** (1), 487–512 (2012).
12. **R. Kannan,** Improved algorithms for integer programming and related lattice problems, in *Proc. 15th Annu. ACM Symp. Theory of Computing, Boston, USA, Apr. 25–27, 1983* (ACM, New York, 1983), pp. 193–206.
13. **U. Fincke** and **M. Pohst,** Improved methods for calculating vectors of short length in a lattice, including a complexity analysis, *Math. Comput.* **44** (170), 463–471 (1985).
14. **N. Gama, P. Q. Nguyen,** and **O. Regev,** Lattice enumeration using extreme pruning, in *Advances in Cryptology — EUROCRYPT 2010* (Proc. 29th Annu. Int. Conf. Theory and Application of Cryptographic Techniques, French Riviera, May 30 – June 3, 2010) (Springer, Heidelberg, 2010), pp. 257–278 (Lect. Notes Comput. Sci., Vol. 6110).
15. **A. K. Lenstra, H. W. Lenstra,** and **L. Lovász,** Factoring polynomials with rational coefficients, *Math. Ann.* **261** (4), 515–534 (1982), DOI: 10.1007/BF01457454.
16. **Y. Chen** and **P. Q. Nguyen,** BKZ 2.0: Better lattice security estimates, in *Advances in Cryptology — ASIACRYPT 2011* (Proc. 17th Int. Conf. Theory and Application of Cryptology and Information Security, Seoul, South Korea, Dec. 4–8, 2011) (Springer, Heidelberg, 2011), pp. 1–20 (Lect. Notes Comput. Sci., Vol. 7073).

17. **C. P. Schnorr,** A hierarchy of polynomial time lattice basis reduction algorithms, *Theor. Comput. Sci.* **53** (2–3), 201–224 (1987), DOI: 10.1016/0304-3975(87)90064-8.
18. **C. P. Schnorr** and **M. Euchner,** Lattice basis reduction: Improved practical algorithms and solving subset sum problems, *Math. Program.* **66**, 181–199 (1994), DOI: 10.1007/BF01581144.
19. **A. Becker, L. Ducas, G. Gama,** and **T. Laarhoven,** New directions in nearest neighbor searching with applications to lattice sieving, in *Proc. 27th Annu. ACM-SIAM Symp. Discrete Algorithms, Arlington, VA, USA, Jan. 10–12, 2016* (SIAM, Philadelphia, PA, 2016), pp. 10–24.
20. **G. Herold, E. Kirshanova,** and **T. Laarhoven,** Speed-ups and time-memory trade-offs for tuple lattice sieving, in *Public-Key Cryptography — PKC 2018* (Proc. 21st IACR Int. Conf. Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, Mar. 25–29, 2018), Pt. I (Springer, Cham, 2018), pp. 407–436 (Lect. Notes Comput. Sci., Vol. 10769).
21. **D. Micciancio** and **P. Voulgaris,** Faster exponential time algorithms for the shortest vector problem, in *Proc. 21st Annu. ACM-SIAM Symp. Discrete Algorithms, Austin, TX, USA, Jan. 17–19, 2010* (SIAM, Philadelphia, PA, 2010), pp. 1468–1480.
22. **P. Q. Nguyen** and **T. Vidick,** Sieve algorithms for the shortest vector problem are practical, *J. Math. Cryptol.* **2** (2), 181–207 (2008), DOI: 10.1515/JMC.2008.009.
23. **X. Pujol** and **D. Stehlé,** Solving the shortest lattice vector problem in time $2^{2.465n}$ (Univ. California, San Diego, 2009) (Cryptology ePrint Archive, Pap. 2009/605). URL: eprint.iacr.org/2009/605 (accessed: 9.10.2023).
24. **D. Aggarwal, D. Dadush, O. Regev,** and **N. Stephens-Davidowitz,** Solving the shortest vector problem in 2^n time using discrete Gaussian sampling, in *Proc. 47th ACM Symp. Theory of Computing, Portland, OR, USA, June 14–17, 2015* (ACM, New York, 2015), pp. 733–742.
25. **E. Doulgerakis, T. Laarhoven,** and **B. de Weger,** Finding closest lattice vectors using approximate Voronoi cells, in *Post-Quantum Cryptography* (Rev. Sel. Pap. 10th Int. Conf. Chongqing, China, May 8–10, 2019) (Springer, Cham, 2019), pp. 3–22 (Lect. Notes Comput. Sci., Vol. 11505).
26. **D. Micciancio** and **P. Voulgaris** A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations, in *Proc. 42nd ACM Symp. Theory of Computing, Cambridge, MA, USA, June 5–8, 2010* (ACM, New York, 2010), pp. 351–358.
27. **D. V. Denisenko** and **M. V. Nikitenkova,** Application of Grover's quantum algorithm for SDES key searching, *Zh. Eksp. Teor. Fiz.* **155** (1), 32–53 (2019) [Russian] [*J. Exp. Teor. Phys.* **128** (1), 25–44 (2019)].
28. **D. V. Denisenko, G. B. Marshalko, M. V. Nikitenkova, V. I. Rudskoi,** and **V. A. Shishkin,** Estimating the complexity of Grover's algorithm for key search of block ciphers defined by GOST R 34.12-2015, *Zh. Eksp. Teor. Fiz.* **155** (4), 645–653 (2019) [Russian] [*J. Exp. Teor. Phys.* **128** (4), 552–559 (2019)].

29. **M. Almazrooie, A. Samsudin, R. Abdullah,** and **K. N. Mutter,** Quantum exhaustive key search with simplified-DES as a case study, *SpringerPlus* **5** (1), 1–19 (2016).
30. **X. Dong, B. Dong,** and **X. Wang,** Quantum attacks on some Feistel block ciphers, *Des. Codes Cryptogr.* **88** (6), 1179–1203 (2020), DOI: 10.1007/s10623-020-00741-y.
31. **P. Frixons, M. Naya-Plasencia,** and **A. Schrottenloher,** Quantum boomerang attacks and some applications, in *Selected Areas in Cryptography* (Proc. 28th Int. Conf., Virtual Event, Sept. 29 – Oct. 1, 2021) (Springer, Cham, 2021), pp. 332–352 (Lect. Notes Comput. Sci., Vol. 13203).
32. **S. Jaques, M. Naehrig, M. Roetteler,** and **F. Virdia,** Implementing Grover oracles for quantum key search on AES and LowMC, in *Advances in Cryptology — EUROCRYPT 2020* (Proc. 39th Annu. Int. Conf. Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020), Pt. II (Springer, Cham, 2020), pp. 280–310 (Lect. Notes Comput. Sci., Vol. 12106).
33. **M. Grassl, B. Langenberg, M. Roetteler,** and **R. Steinwandt,** Applying Grover's algorithm to AES: quantum resource estimates, in *Post-Quantum Cryptography* (Proc. 7th Int. Workshop, Fukuoka, Japan, Feb. 24–26, 2016) (Springer, Cham, 2016), pp. 29–43 (Lect. Notes Comput. Sci., Vol. 9606).
34. **B. Langenberg, H. Pham,** and **R. Steinwandt,** Reducing the cost of implementing the advanced encryption standard as a quantum circuit, *IEEE Trans. Quantum Eng.* **1**, 1–12 (2020).
35. **J. Zou, Z. Wei, S. Sun, X. Liu,** and **W. Wu,** Quantum circuit implementations of AES with fewer qubits, in *Advances in Cryptology — ASIACRYPT 2020* (Proc. 26th Int. Conf. Theory and Application of Cryptology and Information Security, Daejeon, South Korea, Dec. 7–11, 2020), Pt. II (Springer, Cham, 2020), pp. 697–726 (Lect. Notes Comput. Sci., Vol. 12492).
36. **M. R. Albrecht, V. Gheorghiu, E. W. Postlethwaite,** and **J. M. Schanck,** Estimating quantum speedups for lattice sieves, in *Advances in Cryptology — ASIACRYPT 2020* (Proc. 26th Int. Conf. Theory and Application of Cryptology and Information Security, Daejeon, South Korea, Dec. 7–11, 2020), Pt. II (Springer, Cham, 2020), pp. 583–613 (Lect. Notes Comput. Sci., Vol. 12492).
37. **C. Gidney** and **M. Ekerå,** How to factor 2048 bit RSA integers in 8 shours using 20 million noisy qubits, *Quantum* **5**, 433 (2021), DOI: 10.22331/q-2021-04-15-433.
38. **T. Laarhoven, M. Mosca,** and **J. van de Pol,** Finding shortest lattice vectors faster using quantum search, *Des. Codes Cryptogr.* **77** (2–3), 375–400 (2015).
39. **S. Perriello, A. Barenghi,** and **G. Pelosi,** A complete quantum circuit to solve the information set decoding problem, in *Proc. 2021 IEEE Int. Conf. Quantum Computing and Engineering, Broomfield, CO, USA, Oct. 17–22, 2021* (IEEE Comput. Soc., Los Alamitos, CA, 2021), pp. 366–377.

40. A. O. Bakharev, Estimates of implementation complexity for quantum cryptanalysis of post-quantum lattice-based cryptosystems, *Diskretn. Anal. Issled. Oper.* **30** (3), 5–42 (2023) [Russian] [*J. Appl. Ind. Math.* **17** (3), 459–482 (2023)].
41. L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proc. 28th ACM Symp. Theory of Computing, Philadelphia, PA, USA, May 22–24, 1996* (ACM, New York, 1996), pp. 212–219.
42. M. A. Nielsen and **I. L. Chuang,** *Quantum computation and quantum information* (Camb. Univ. Press, Cambridge, 2010).
43. A. Yu. Kitaev, A. Kh. Shen, and **M. N. Vyalyi,** Classical and quantum computations (MTsNMO; CheRo, Moscow, 1999).
44. M. Boyer, G. Brassard, P. Høyer, and **A. Tapp,** Tight bounds on quantum searching, *Fortschr. Phys.* **46** (4–5), 493–505 (1998), DOI: `10.1145/237814.237866`.
45. C. Moore and **M. Nilsson,** Parallel quantum computation and quantum codes, *SIAM J. Comput.* **31** (3), 799–815 (2001).
46. C. Chen, O. Danba, J. Hoffstein, [et al.], NTRU Algorithm Specifications and Supporting Documentation (Eindh. Univ. Technol., Eindhoven, 2019), URL: `ntru.org/f/ntru-20190330.pdf` (accessed: 9.10.2023).
47. J.-P. D'Anvers, A. Karmakar, S. S. Roy, and **F. Vercauteren,** SABER: Mod-LWR Based KEM (KU Leuven, Leuven, 2017), URL: `esat.kuleuven.be/cosic/pqcrypto/saber/files/saberspecround1.pdf` (accessed: 9.10.2023).
48. R. Avanzi, J. Bos, L. Ducas, [et al.], CRYSTALS-Kyber Algorithm Specifications and Supporting Documentation (Cent. Wiskd. Inform., Amsterdam, 2021), URL: `cryptojedi.org/papers/kybernist-20171130.pdf` (accessed: 9.10.2023).

Aleksandr O. Bakharev

Received June 27, 2023
Revised November 27, 2023
Accepted March 22, 2024