Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)
КАФЕДРА «Информационная безопасность» (ИУ8)

Курсовая работа
Факторизация целых чисел с использованием сублинейных ресурсов на сверхпроводниковом квантовом процессоре

по дисциплине «Криптографические методы защиты информации»

Студент ИУ8-104 (Группа) — Н. В. Железцов (И. О. Фамилия) — _____ (Подпись, дата)

Преподаватель — А. Е. Жуков (И. О. Фамилия) — _____ (Подпись, дата)

Оценка: _____

Москва, 2025 г.

РЕФЕРАТ

Алгоритм Шора серьёзно поставил под вопрос безопасность информации, основанную на криптосистемах с открытым ключом. Однако для взлома широко используемой схемы RSA-2048 требуются миллионы физических кубитов, что значительно превышает текущие технические возможности. Здесь мы сообщаем об универсальном квантовом алгоритме факторизации целых чисел, объединяющем классическую редукцию базиса решетки с квантовым алгоритмом приближённой оптимизации (QAOA — quantum approximate optimization algorithm). Количество требуемых кубитов равно $O(\log N/\log\log N)$, что сублинейно относительно битовой длины целого числа N, делая этот алгоритм самым экономичным по числу кубитов алгоритмом факторизации на сегодняшний день. Мы экспериментально тестируем алгоритм, факторизуя целые числа размером до 48 бит с помощью 10 сверхпроводящих кубитов, что является наибольшим целым числом, факторизованным на квантовом устройстве. Мы оцениваем, что квантовая схема с 372 физическими кубитами и глубиной в несколько тысяч операций необходима для того, чтобы при помощи нашего алгоритма бросить вызов RSA-2048. Наше исследование демонстрирует значительные перспективы для ускорения применения текущих шумных квантовых компьютеров и прокладывает путь к факторизации больших целых чисел, имеющих реальное криптографическое значение.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Квантовые вычисления вступили в эпоху "шумных"квантовых устройств промежуточного масштаба (NISQ — noisy intermediate scale quantum) [1; 2]. Важной задачей эпохи NISQ является демонстрация того, что устройства NISQ могут превзойти классические компьютеры при решении задач с практической значимостью, то есть достижение практического квантового преимущества. Алгоритмы, требующие минимальных ресурсов и использующие ограниченное число доступных кубитов и ограниченную глубину схем для решения задач, сложных для классических вычислений, имеют особую важность. Вариационные квантовые алгоритмы, использующие гибридную схему вычислений «классические и квантовые вычисления», обладают значительным потенциалом для получения значимого квантового преимущества в эпоху NISQ [2—6]. Одним из таких алгоритмов является квантовый алгоритм приближённой оптимизации (QAOA — quantum approximate optimization algorithm) [5], первоначально предложенный для решения задач на собственные значения, который впоследствии широко применялся в различных областях, таких как химическое моделирование [7; 8], машинное обучение [9], а также инженерные приложения [10; 11].

Факторизация целых чисел является одной из важнейших основ современной информационной безопасности [12]. Экспоненциальное ускорение факторизации алгоритмом Шора [13] является выдающимся примером превосходства квантовых вычислений. Однако выполнение алгоритма Шора на отказоустойчивом квантовом компьютере требует значительных ресурсов [14; 15]. На сегодняшний день наибольшее целое число, факторизованное алгоритмом Шора на существующих квантовых системах, это число 21 [16—18]. Альтернативно, факторизация целых чисел может быть сведена к задаче оптимизации, решаемой посредством адиабатических квантовых вычислений (AQC — adiabatic quantum computation) [19—22] или QAOA [23]. Более крупные числа были факторизованы этими методами на различных физических системах [24—27]. Максимальные числа, факторизованные на данный момент, включают число 291311 (19 бит) в системе NMR (nuclear magnetic resonance) [26], 249919 (18 бит) на квантовом отжигателе D-Wave [25], 1099551473989 (41 бит) на сверхпроводя-

щем устройстве [27]. Однако следует отметить, что некоторые из факторизованных чисел были специально подобраны как числа с особыми структурами [28], поэтому наибольшее число, факторизованное универсальным методом на реальной физической системе, на сегодняшний день составляет 249919 (18 бит).

В данной работе мы предлагаем универсальный квантовый алгоритм факторизации целых чисел, требующий лишь сублинейные квантовые ресурсы. Алгоритм основан на классическом алгоритме Шнорра [29; 30], использующем редукцию базиса решётки для факторизации целых чисел. Мы используем QAOA для оптимизации наиболее трудоёмкой части алгоритма Шнорра, что ускоряет общее время факторизации. Для целого числа N, имеющего m бит, количество требуемых кубитов в нашем алгоритме составляет $O(m/\log m)$, что сублинейно относительно битовой длины числа N. Это делает наш алгоритм наиболее экономным по числу кубитов по сравнению с существующими алгоритмами, включая алгоритм Шора. С использованием данного алгоритма нами успешно факторизованы числа 1961 (11 бит), 48567227 (26 бит) и 261980999226229 (48 бит) с использованием, соответственно, 3, 5 и 10 кубитов на сверхпроводящем квантовом процессоре. Число в 48 бит (261980999226229) также является наибольшим целым числом, факторизованным универсальным методом на реальном квантовом устройстве. Далее мы оцениваем квантовые ресурсы, необходимые для факторизации RSA-2048. Согласно нашим расчётам, квантовая схема с 372 физическими кубитами и глубиной порядка нескольких тысяч операций необходима для факторизации RSA-2048 даже в самой простой одномерной системе. Подобный масштаб квантовых ресурсов, вероятно, станет достижимым на устройствах NISQ в ближайшем будущем.

ОСНОВНАЯ ЧАСТЬ

1 Структура алгоритма

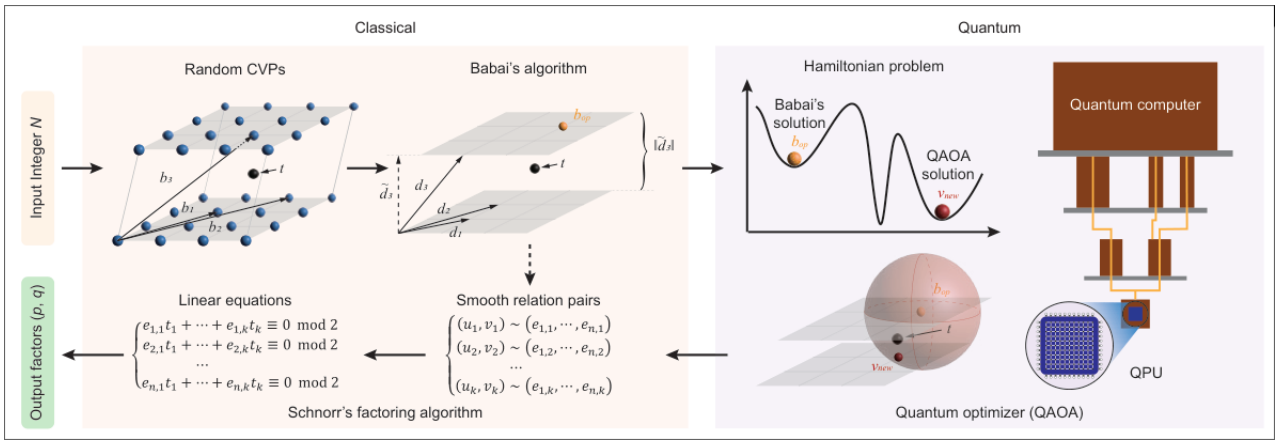

Рисунок 1 – Порядок работы квантового алгоритма факторизации целых чисел с сублинейными ресурсами (SQIF). Алгоритм использует гибридную «классическую+квантовую» структуру, в которой квантовый оптимизатор QAOA применяется для оптимизации классического алгоритма факторизации Шнорра. Сначала задача предварительно обрабатывается как задача ближайшего вектора (CVP) на решётке. Затем квантовый компьютер работает в качестве оптимизатора, уточняя классические векторы, полученные алгоритмом Бабаи; на этом этапе находится решение CVP более высокого качества (то есть более близкое). Оптимизированные результаты передаются обратно в процедуру алгоритма Шнорра. После пост-обработки в конце выводятся множители p и q

Порядок работы квантового алгоритма факторизации целых чисел с сублинейными ресурсами (SQIF — sublinear-resource quantum integer factorization) представлен на рис. 1, который, по существу, представляет собой гибридную «классическую+квантовую» структуру. Основная идея состоит в использовании квантового оптимизатора QAOA для оптимизации самой ресурсоёмкой части алгоритма Шнорра, что, в результате, повышает общую эффективность процесса факторизации. Как показано на левой панели рис. 1, алгоритм Шнорра включает два существенных шага: поиск достаточного количества пар гладких отношений (для краткости, sr-пары) и решение полученной системы линейных уравнений. Как правило, поиск sr-пар является самой важной и трудоёмкой частью алгоритма, тогда как решение системы уравнений может быть выполнено за полиномиальное время. В алгоритме Шнорра [31] задача sr-пар преобразуется в задачу ближайшего вектора (CVP — closest vector problem) на решётке и решается посредством алгоритмов редукции базиса решётки, та-

ких как алгоритм Бабаи [32]. Поскольку CVP является известной NP-трудной задачей [33], мы можем рассчитывать только на приближённое, а не точное решение CVP за полиномиальное или другое приемлемое время. Вероятность получения sr-пары пропорциональна качеству решения CVP [29]. То есть чем ближе вектор-решение CVP, тем эффективнее поиск sr-пар. Исходя из изложенного, мы предлагаем схему, в которой QAOA дополнительно оптимизирует решение CVP, полученное алгоритмом Бабаи. Полный процесс алгоритма SQIF представлен в подробных примерах в [31]. В дальнейшем мы сосредоточимся на квантовых этапах алгоритма.

Мы объединяем алгоритм Бабаи с QAOA для решения CVP на решётке. Пусть дана решётка Λ с базисом $B = [b_1, \ldots, b_n] \in R^{(n+1)\times n}$ и целевой вектор $t \in R^{n+1}$. Алгоритм Бабаи находит вектор $b_{op} \in \Lambda$, приближённо ближайший к t, в два шага. Сначала выполняется LLL-редукция с параметром δ для заданного базиса B. В результате получаем LLL-редуцированный базис $D = [d_1, \ldots, d_n]$ и соответствующий ортогональный базис Грама–Шмидта $\widetilde{D} = [\widetilde{d}_1, \ldots, \widetilde{d}_n]$. Второй шаг — «уменьшение размера» целевого вектора t с использованием LLL-редуцированного базиса. Так получаем приближённый ближайший вектор

$$b_{op} = \left(b_{op}^1, \ldots, b_{op}^{n+1}\right)' = \sum_{i=1}^{n} c_i \, d_i, \tag{1}$$

где коэффициент $c_i = \lceil \mu_i \rfloor = \left\lceil \langle d, \widetilde{d}_i \rangle / \langle \widetilde{d}_i, \widetilde{d}_i \rangle \right\rfloor$ получается округлением к ближайшему целому коэффициента Грама–Шмидта μ_i. Здесь следует отметить, что функция округления к ближайшему значению выполняет лишь одно приближение за раз. Фактически, если значения двух функций округления можно учесть одновременно, удаётся получить решение более высокого качества [31]. Однако этот процесс экспоненциально увеличивает объём классических операций, что неприемлемо для классического компьютера. Мы применяем идею квантовых вычислений, используя суперпозицию кубитов для одновременного кодирования значений коэффициентов, полученных двумя функциями округления. Затем мы формулируем задачу оптимизации, основываясь на евклидовом расстоянии между новым вектором решётки и целевым вектором. Подробности построения приведены ниже.

Пусть v_{new} — новый вектор, полученный случайным выбором $x_i \in \{0, \pm 1\}$ на коэффициенте c_i, удовлетворяющий соотношению

$$v_{new} \;=\; \sum_{i=1}^{n}(c_i + x_i)\, d_i \;=\; \sum_{i=1}^{n} x_i\, d_i + b_{op}. \tag{2}$$

Мы определяем функцию потерь задачи оптимизации следующим образом:

$$F(x_1, \ldots, x_n) \;=\; \left\|t - v_{new}\right\|^2 \;=\; \left\| t - \sum_{i=1}^{n} x_i\, d_i - b_{op}\right\|^2. \tag{3}$$

Значение функции $\|t - v_{new}\|^2$ представляет собой квадрат евклидова расстояния от нового вектора до целевого вектора. Чем меньше значение функции потерь, тем ближе новый вектор к целевому вектору t и тем выше качество решения. Когда все переменные $x_{i,i=1,\ldots,n}$ принимают нулевые значения, получаем оптимальное решение, выдаваемое алгоритмом Бабаи.

Отображая переменную x_i на операторы Паули-Z, можно построить гамильтониан задачи, соответствующий уравнению (3), в виде

$$H_C \;=\; \left\|t - \sum_{i=1}^{n} \hat{x}_i\, d_i - b_{op}\right\|^2 \;=\; \sum_{j=1}^{n+1}\left|t_j - \sum_{i=1}^{n} \hat{x}_i\, d_{i,j} - b_{op}^{j}\right|^2, \tag{4}$$

где $\hat{x}_i$ — квантовый оператор, сопоставленный базису Паули-Z согласно правилам однокубитного кодирования (см. [31]).

В таком случае число кубитов, необходимых для квантовой процедуры оптимизации алгоритма Бабаи, равно размерности решётки. Согласно анализу из [31], эта размерность удовлетворяет $n \sim 2c \log N / \log\log N$, где c — параметр решётки, близкий к 1. Следовательно, чтобы факторизовать m-битное целое число N, наш алгоритм требует $O(m/\log m)$ кубитов — сублинейную величину по m, в то время как алгоритм Шора нуждается в $O(m)$ кубитах [13], а метод таблицы произведений — в $O(m^2)$ кубитах [25]. Тем самым наш алгоритм является самым экономичным по числу кубитов на сегодняшний день и первым универсальным квантовым алгоритмом факторизации с сублинейными квантовыми ресурсами.

2 Эксперимент и результаты

Мы демонстрируем работу алгоритма, экспериментально факторизуя три целых числа на сверхпроводящем квантовом процессоре, где выбраны десять кубитов и девять соединителей, расположенных в топологии цепочки. Все кубиты и соединители — это настраиваемые по частоте сверхпроводящие кубиты (transmon — трансмон); однокубитные вращения вокруг осей x или y сферы Блоха реализуются подачей управляющих сигналов, в которых информация затвора закодирована в амплитуде и фазе СВЧ-импульсов. Виртуальные z-затворы используются для выполнения вращений вокруг оси z. Двухкубитные затворы с контролируемой файзой (CZ) реализуются за счёт обмена совместными состояниями $|11\rangle$ и $|02\rangle$ (или $|20\rangle$) соседних кубитов, когда активируется взаимодействие, передаваемое через соединитель [34]. Параллельные измерения производительности кросс-энтропии (XEB — cross-entropy benchmarking) дают средние достоверности порядка 99,9 % для однокубитных вращений и 99,5 % для CZ-затворов. Подробности экспериментальной установки и характеристики процессора приведены в [31].

Мы факторизуем 11-битное число 1961, 26-битное число 48567227 и 48-битное число 261980999226229, используя соответственно 3, 5 и 10 сверхпроводящих кубитов. Ниже показан процесс получения одной sr-пары квантовым методом в каждой серии экспериментов. Вычисления остальных sr-пар аналогичны и выполняются численно; все sr-пары и соответствующие системы линейных уравнений приведены в [31].

Топология ZZ-слагаемых в гамильтониане задачи согласно уравнению (4) представляет собой K_n — полный граф порядка n [31]. Пример для 10 кубитов показан на рис. 2B. Чтобы реализовать гамильтониан типа K_n на линейной 1D-цепочке физических кубитов, мы используем маршрутный метод, основанный на классическом параллельном алгоритме пузырьковой сортировки: все-ко-всем взаимодействия кубитов отображаются на взаимодействия ближайших друг к другу двух кубитов с помощью сложных swap сетей, как показано на рис. 2D. Этот маршрутный метод является оптимальным и добавляет лишь линейное увеличение глубины схемы. SWAP сети компилируются в затворы рис. (рис. 2E), которые могут исполняться непосредственно на квантовом процессоре. Отметим маленький приём: чередование блоков ZZ-SWAP вверх-вниз в чётных и нечётных слоях сетей позволяет сократить линейное число затворов H.

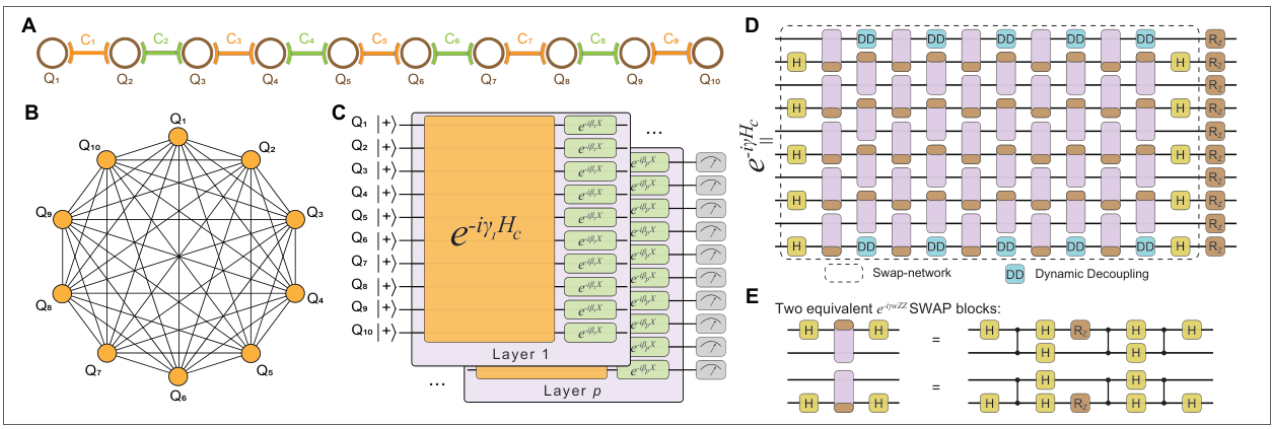

Рисунок 2 – Экспериментальная установка и схема QAOA алгоритма SQIF. **A**. Десять выбранных кубитов на сверхпроводящем квантовом процессоре; каждый кубит связан с ближайшими соседями через настраиваемые по частоте соединители. **B**. Родная топология взаимодействий гамильтониана задачи для случая факторизации на 10 кубитах, отображённая в линейную цепочку, показанную в A. **C**. Схема p-слойного QAOA. Все кубиты инициализируются в состояние $|+\rangle$, далее следуют p слоёв попеременного действия гамильтониана задачи (оранжевый) и смешивающего гамильтониана (зелёный) и завершающие измерения популяций (серый). Обратите внимание, что вариационные параметры $\{\gamma, \beta\}$ различны для каждого слоя. **D**. Маршрутная схема для отображения все-к-о-всем гамильтониана 10 кубитов на линейную топологию ближайших соседей; построена «кирпичной» сеткой из двух одинаковых блоков SWAP с двумя слоями затворов H (Hadamard) в начале и конце, за которыми следует слой затворов $R_z(\theta)$. Здесь угол вращения опущен. Глубина схемы пропорциональна числу задействованных кубитов. **E**. Подробная компиляция квантовой схемы в затворы сверхпроводящего квантового процессора.

QAOA находит приближённое основное состояние гамильтониана путём обновления параметров (рис. 2C; подробности см. [31]). Оптимизацию параметров QAOA можно проиллюстрировать через ландшафт функции энергии $E(\gamma, \beta)$. Сравнение теоретического и экспериментального ландшафтов служит качественной диагностикой применимости QAOA на реальном оборудовании. Для гиперпараметра $p = 1$ ландшафт энергии как функция (γ, β) изображён в 3D графике на рис. 3. Значения энергии нормированы по $E^* = (E - E_{\min})/(E_{\max} - E_{\min})$. На рис. 3 показаны симуляция без шума (слева) и эксперимент (справа) для случаев 3, 5 и 10 кубитов. Различные цвета пикселей обозначают разные значения функции; поверх наложена траектория сходимости классического оптимизатора (красная кривая). Для оптимизации используется метод градиентного спуска по модели, показавший хорошую работу как в теории, так и на практике. Алгоритм сходится в область глобального минимума

менее чем за 10 шагов во всех трёх случаях. Хотя пути сходимости эксперимента отличаются от теоретических, они достигают оптимума за сравнимое число шагов. Это свидетельствует об устойчивости алгоритма к определённому уровню шума.

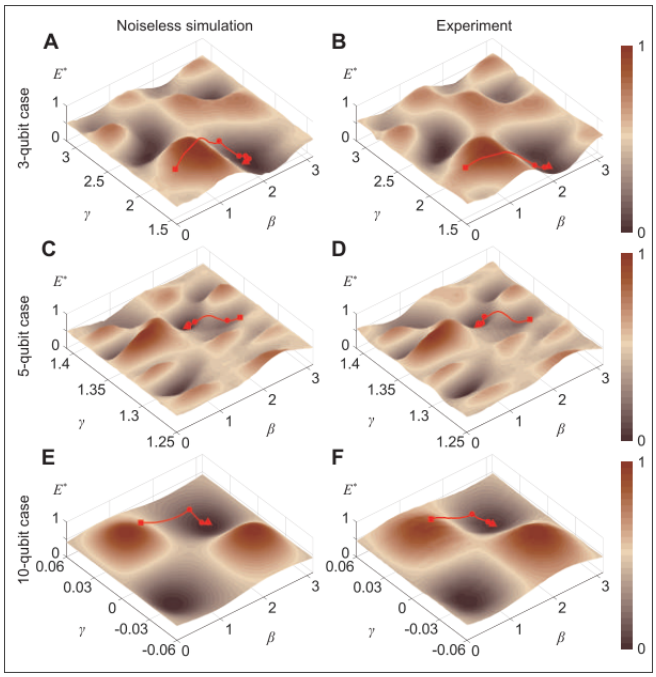

Рисунок 3 – Энергетические ландшафты и траектории сходимости QAOA при $p = 1$. **A**, **B** — численные и экспериментальные ландшафты для случая 3 кубитов; **C**, **D** — для 5 кубитов; **E**, **F** — для 10 кубитов. В каждой экспериментальной группе оценены 41×41 комбинаций (γ, β), равномерно распределённых по сетке в подобласти всего двумерного пространства параметров. Для каждой точки сетки математическое ожидание энергии вычисляется по результатам 30 000 повторов схемы. Сравнение экспериментальных и численных ландшафтов демонстрирует отчётливое соответствие характерных особенностей. Наложенная траектория оптимизации (красная; стартует из квадратного маркера и сходится в треугольник) показывает способность классического оптимизатора находить оптимальные параметры.

В QAOA основная работа квантового компьютера состоит в подготовке квантовых состояний с заданными вариационными параметрами. Теоретически производительность QAOA улучшается с ростом глубины гиперпараметра p. Однако с увеличением глубины накапливаются ошибки, и выгода может нивелироваться. Ниже приведены результаты процессора при оптимальных (β, γ). Мы демонстрируем слои QAOA до $p = 3$ для 3- и 5-кубитных случаев и односоставную QAOA ($p = 1$) для 10 кубитов. Для 10 кубитов также выполнен запуск при $p = 3$; результат лучше случайного угадывания, но хуже, чем при $p = 1$ [31]. На рис. 4A-C видно, что вероятность целевого состояния (красная пунктирная рамка) растёт с увеличением p. Хотя рост меньше теоретическо-

го, он хорошо согласуется с шумовой симуляцией. Аналогичные результаты получены в 5-кубитном эксперименте (рис. 4D-F). Для 10 кубитов при $p = 1$ результаты показаны на рис. 4G; для наглядности отображены 120 наиболее вероятных состояний по теории. Теоретическая вероятность целевого состояния равна 0.02 (наибольшая), экспериментальная около 0.008, что близко к шумовому значению 0.009. Экспериментальные результаты значительно превосходят случайное угадывание 0.001, то есть вычислительный выигрыш QAOA остаётся существенным. Более того, форма распределения вероятностей симметрична относительно симуляционных данных, что также подтверждает хорошее совпадение эксперимента с теорией.

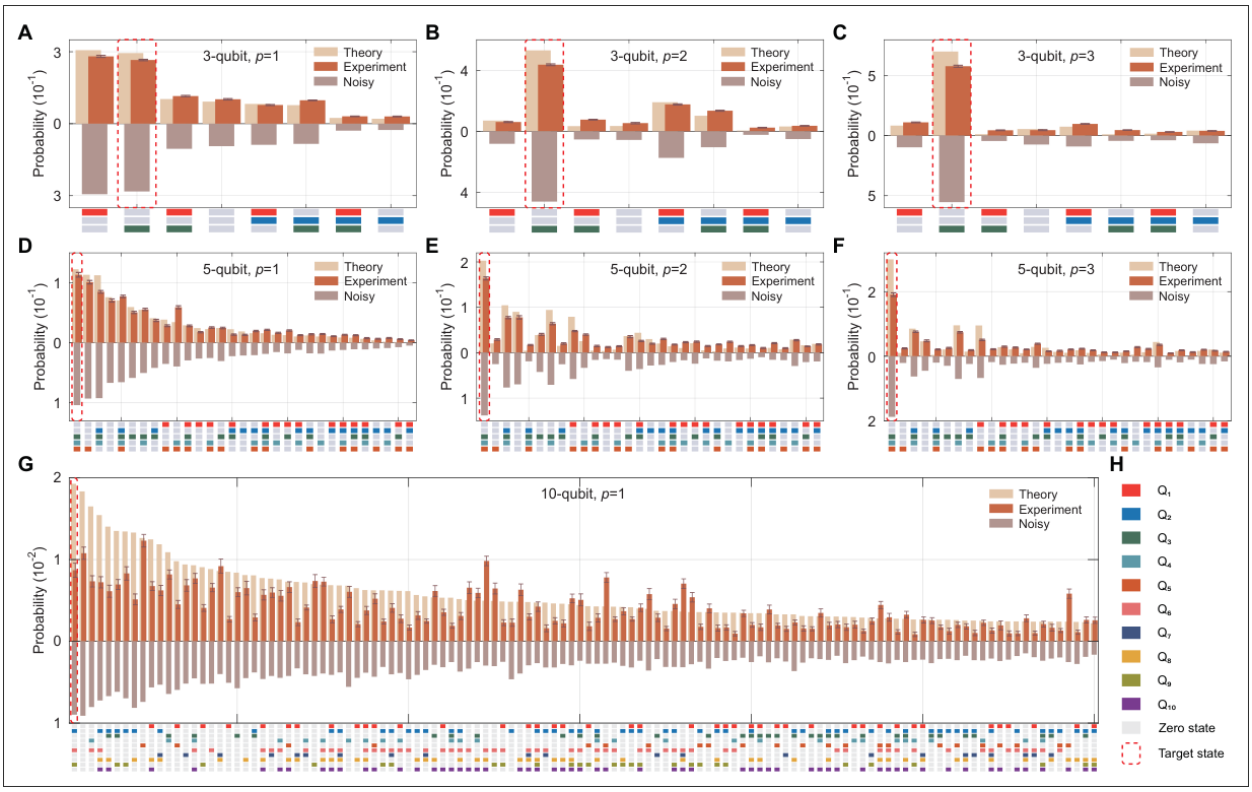

Рисунок 4 – Экспериментальная производительность QAOA для трёх случаев факторизации. **A**–**C**. Производительность QAOA для 3-кубитного случая при $p = 1$, $p = 2$ и $p = 3$ соответственно. **D**–**F**. Производительность QAOA для 5-кубитного случая при $p = 1$, $p = 2$ и $p = 3$ соответственно. **G**. Производительность QAOA при $p = 1$ для 10-кубитного случая. Экспериментальные результаты, показанные оранжевым цветом, усреднены по 20 повторам эксперимента; штрихи ошибок отображают доверительный интервал в одном стандартном отклонении. Для сравнения также приведены теоретические (жёлтый) и шумовые при уровне 0.01 (серо-тауповый) результаты. Видно, что все три группы экспериментальных данных на сверхпроводящем квантовом процессоре хорошо согласуются с теорией и шумовой моделью 0.01. **H**. Обозначения цветовых блоков, соответствующих базисным состояниям различных кубитов в подписи оси x.

3 Оценка квантовых ресурсов

Ниже приведены квантовые ресурсы, необходимые для того чтобы поставить под угрозу некоторые реальные числа RSA, основываясь на алгоритме SQIF, который описан в данной работе. Основные упоминаемые ресурсы — число кубитов и глубина квантовой схемы одного слоя QAOA. Как правило, квантовые схемы нельзя запускать на квантовых устройствах напрямую, поскольку их проект не учитывает топологию связей реальных физических систем. Исполнение часто требует дополнительных ресурсов — вспомогательных кубитов и увеличения глубины схем. Мы рассмотрели ресурсы для трёх типовых топологий: полностью связанной системы (K_n), двумерной решётки (2DSL — 2D-lattice system) и одномерной линейной цепочки (LNN — 1D-chain system). Мы показываем на конкретных схемах, что встраивание не требует лишних кубитов, а глубина одного слоя QAOA остается $O(n)$ во всех трёх случаях. Следовательно, для факторизации целых чисел нашим алгоритмом необходимы сублинейные квантовые ресурсы. Возьмём RSA-2048 в качестве примера, в этом случае требуемое число кубитов $n = 2 \cdot 2048/\log 2048 \approx 372$. Глубина одного слоя QAOA составляет 1118 для топологии K_n, 1139 для 2DSL и 1490 для самой простой LNN, что достижимо для устройств NISQ в ближайшем будущем. Квантовые ресурсы для разных длин чисел RSA приведены в таблице 1. Подробный анализ дан в [31].

Таблица 1 – Оценка ресурсов для чисел RSA. Основные квантовые ресурсы включают число кубитов и глубину квантовой схемы QAOA с одной итерацией в трёх типовых топологиях: полностью связанной системе (K_n), двумерной решётке (2DSL) и линейной цепочке (LNN). Результаты получены без учёта нативной компиляции базового модуля ZZ (или базового модуля ZZ-SWAP) в конкретной физической системе.

RSA number	Qubits	K_n-depth	2DSL-depth	LNN-depth
RSA-128	37	113	121	150
RSA-256	64	194	204	258
RSA-512	114	344	357	458
RSA-1024	205	617	633	822
RSA-2048	372	1118	1139	1490

ЗАКЛЮЧЕНИЕ

Проблема факторизации целых чисел является краеугольным камнем безопасности широко применяемой сегодня криптографии с открытым ключом RSA. В этой работе мы предложили общий квантовый алгоритм факторизации целых чисел, основанный на классическом методе редукции базиса решётки. Для факторизации m-битного целого числа N алгоритму требуется $O(m/\log m)$ кубитов, что составляет сублинейную величину относительно битовой длины N. Этот квантовый алгоритм факторизации использует меньше всего кубитов по сравнению с предыдущими методами, включая алгоритм Шора. Мы продемонстрировали принцип факторизации на сверхпроводящем квантовом процессоре. 48-битное число 261980999226229, факторизованное в нашей работе, является к настоящему времени наибольшим целым числом, разложенным универсальным методом на реальной квантовой системе. Мы проанализировали квантовые ресурсы, необходимые для факторизации RSA-2048 в квантовых системах с тремя типовыми топологиями. Обнаружено, что квантовая схема с 372 физическими кубитами и глубиной в несколько тысяч операций необходима, чтобы бросить вызов RSA-2048 даже в самой простой линейной цепочке. Такой масштаб квантовых ресурсов, вероятно, будет достигнут на устройствах NISQ в ближайшем будущем. Следует отметить, что квантовое ускорение алгоритма остаётся неясным из-за неоднозначной сходимости QAOA. Тем не менее идея оптимизации процедуры «уменьшения размера» в алгоритме Бабаи при помощи QAOA может использоваться как подпрограмма в широком классе распространённых алгоритмов редукции базиса решётки и, далее, способствовать анализу криптографических задач, устойчивых к квантовому взлому, основанных на решётках.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Preskill J. Quantum computing in the NISQ era and beyond // Quantum. — 2018. — Т. 2. — С. 79.

2. Quantum supremacy using a programmable superconducting processor / F. Arute [и др.] // Nature. — 2019. — Т. 574. — С. 505.

3. Variational quantum algorithms / M. Cerezo [и др.] // Nature Reviews Physics. — 2021. — Т. 3. — С. 625.

4. A variational eigenvalue solver on a photonic quantum processor / A. Peruzzo [и др.] // Nature Communications. — 2014. — Т. 5. — С. 4213.

5. Farhi E., Goldstone J., Gutmann S. A quantum approximate optimization algorithm. — 2014. — arXiv:1411.4028.

6. Variational quantum attacks threaten Advanced Encryption Standard based symmetric cryptography / Z. Wang [и др.] // Science China Information Sciences. — 2022. — Т. 65. — С. 1.

7. Quantum computational chemistry / S. McArdle [и др.] // Reviews of Modern Physics. — 2020. — Т. 92. — С. 015003.

8. Wei S., Li H., Long G. A full quantum eigensolver for quantum chemistry simulations // Research. — 2020. — С. 1486934.

9. Quantum machine learning / J. Biamonte [и др.] // Nature. — 2017. — Т. 549. — С. 195.

10. Quantum approximate optimization algorithm for MaxCut: A fermionic view / Z. Wang [и др.] // Physical Review A. — 2018. — Т. 97. — С. 022304.

11. Quantum approximate optimization of non-planar graph problems on a planar superconducting processor / M. P. Harrigan [и др.] // Nature Physics. — 2021. — Т. 17. — С. 332.

12. Rivest R. L., Shamir A., Adleman L. A method for obtaining digital signatures and public-key cryptosystems // Communications of the ACM. — 1978. — Т. 21, № 2. — С. 120—126.

13. Shor P. Algorithms for quantum computation: discrete logarithms and factoring // Proceedings of the 35th Annual Symposium on Foundations of Computer Science. — 1994. — С. 124—134.

14. Gidney C., Ekera M. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits // Quantum. — 2021. — Т. 5. — С. 433.

15. Gouzien E., Sangouard N. Factoring 2048-bit RSA integers in 177 days with 13 436 qubits and a multimode memory // Physical Review Letters. — 2021. — Т. 127. — С. 140503.

16. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance / L. M. Vandersypen [и др.] // Nature. — 2001. — Т. 414. — С. 883.

17. Realization of a scalable Shor algorithm / T. Monz [и др.] // Science. — 2016. — Т. 351. — С. 1068.

18. Experimental realization of Shor's quantum factoring algorithm using qubit recycling / E. Martin-Lopez [и др.] // Nature Photonics. — 2012. — Т. 6. — С. 773.

19. A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem / E. Farhi [и др.] // Science. — 2001. — Т. 292. — С. 472.

20. Schaller G., Schützhold R. The role of symmetries in adiabatic quantum algorithms // Quantum Information & Computation. — 2010. — Т. 10. — С. 109.

21. Integer factorization using stochastic magnetic tunnel junctions / W. A. Borders [и др.] // Nature. — 2019. — Т. 573. — С. 390.

22. Adiabatic quantum algorithm for factorization with growing minimum energy gap / B. Yan [и др.] // Quantum Engineering. — 2021. — Т. 3. — e59.

23. Variational quantum factoring / E. Anschuetz [и др.] // International Workshop on Quantum Technology and Optimization Problems. — Springer, 2019. — С. 74—85.

24. Experimental adiabatic quantum factorization under ambient conditions based on a solid-state single spin system / K. Xu [и др.] // Physical Review Letters. — 2017. — Т. 118. — С. 130504.

25. Quantum annealing for prime factorization / S. Jiang [и др.] // Scientific Reports. — 2018. — Т. 8. — С. 17667.

26. High-fidelity adiabatic quantum computation using the intrinsic Hamiltonian of a spin system: Application to the experimental factorization of 291311 / Z. Li [и др.]. — 2017. — arXiv:1706.08061.

27. Analyzing the performance of variational quantum factoring on a superconducting quantum processor / A. H. Karamlou [и др.] // npj Quantum Information. — 2021. — Т. 7. — С. 155.

28. Mosca M., Verschoor S. R. Factoring semi-primes with (quantum) SAT-solvers // Scientific Reports. — 2022. — Т. 12. — С. 6723.

29. Schnorr C. P. Factoring integers by CVP algorithms // Number Theory and Cryptography / под ред. M. Jacobson. — Springer, 2013. — С. 73—93.

30. Schnorr C. P. Fast factoring integers by SVP algorithms, corrected. — 2021. — Cryptology ePrint Archive.

31. Supplementary materials. — 2023. — See supplementary materials.

32. Babai L. On Lovász' lattice reduction and the nearest lattice point problem // Combinatorica. — 1986. — Т. 6. — С. 1.

33. Micciancio D. The hardness of the closest vector problem with preprocessing // IEEE Transactions on Information Theory. — 2001. — Т. 47. — С. 1212.

34. Digital quantum simulation of Floquet symmetry-protected topological phases / X. Zhang [и др.] // Nature. — 2022. — Т. 607. — С. 468.

35. Lenstra A. K., Lenstra H. W., Lovász L. Factoring polynomials with rational coefficients // Mathematische Annalen. — 1982. — Т. 261. — С. 515.

36. Ajtai M., Kumar R., Sivakumar D. A sieve algorithm for the shortest lattice vector problem // Proceedings of STOC '01. — 2001. — С. 601—610.

37. Schnorr C.-P., Euchner M. Lattice basis reduction: Improved practical algorithms and solving subset sum problems // Mathematical Programming. — 1994. — Т. 66. — С. 181.

38. Fincke U., Pohst M. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis // Mathematics of Computation. — 1985. — Т. 44. — С. 463.

39. Schnorr C.-P., Hörner H. H. Attacking the Chor–Rivest cryptosystem by improved lattice reduction // Proceedings of EUROCRYPT '95. — Springer, 1995. — С. 1—12.

40. Gama N., Nguyen P. Q., Regev O. Lattice enumeration using extreme pruning // Proceedings of EUROCRYPT '10. — Springer, 2010. — С. 257—278.

41. Schnorr C. Factoring integers and computing discrete logarithms via Diophantine approximation // Proceedings of EUROCRYPT '91. — 1991. — С. 281—293.

42. Cassels J. W. S. An Introduction to the Geometry of Numbers. — Springer, 2012.

43. Kabatiansky G. A., Levenshtein V. I. On bounds for packings on a sphere and in space // Problemy Peredachi Informatsii. — 1978. — Т. 14. — С. 3.

44. Digital simulation of non-Abelian anyons with 68 programmable superconducting qubits / S. Xu [и др.]. — 2022. — arXiv:2211.09802.

45. Scalable evaluation of quantum-circuit error loss using Clifford sampling / Z. Wang [и др.] // Physical Review Letters. — 2021. — Т. 126. — С. 080501.

46. Efficient Z gates for quantum computing / D. C. McKay [и др.] // Physical Review A. — 2017. — Т. 96. — С. 022330.

47. Experimental quantum adversarial learning with programmable superconducting qubits / W. Ren [и др.]. — 2022. — arXiv:2204.01738.

48. Using models to improve optimizers for variational quantum algorithms / K. J. Sung [и др.] // Quantum Science and Technology. — 2020. — Т. 5. — С. 044008.

49. Convergence properties of the Nelder–Mead simplex method in low dimensions / J. C. Lagarias [и др.] // SIAM Journal on Optimization. — 1998. — Т. 9. — С. 112.

50. Broyden C. G. The convergence of a class of double-rank minimization algorithms: general considerations // IMA Journal of Applied Mathematics. — 1970. — Т. 6. — С. 76.

51. Liu D. C., Nocedal J. On the limited memory BFGS method for large scale optimization // Mathematical Programming. — 1989. — Т. 45. — С. 503.

52. Quantum approximate optimization of the long-range Ising model with a trapped-ion quantum simulator / G. Pagano [и др.] // Proceedings of the National Academy of Sciences. — 2020. — Т. 117. — С. 25396.

53. Takahashi Y., Kunihiro N., Ohta K. The quantum Fourier transform on a linear nearest neighbor architecture // Quantum Information & Computation. — 2007. — Т. 7. — С. 383.

54. Kutin S. A. Shor's algorithm on a nearest-neighbor machine. — 2006. — arXiv:quant-ph/0609001.

55. Cheung D., Maslov D., Severini S. Translation techniques between quantum circuit architectures // Workshop on Quantum Information Processing. — 2007.

56. An efficient method to convert arbitrary quantum circuits to ones on a linear nearest neighbor architecture / Y. Hirata [и др.] // Proceedings of ICQNM '09. — IEEE, 2009. — С. 26—33.

57. Saeedi M., Wille R., Drechsler R. Synthesis of quantum circuits for linear nearest neighbor architectures // Quantum Information Processing. — 2011. — Т. 10. — С. 355.

58. Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits / R. Wille [и др.] // Proceedings of ASP-DAC '16. — IEEE, 2016. — С. 292—297.

59. Farghadan A., Mohammadzadeh N. Quantum circuit physical design flow for 2D nearest-neighbor architectures // International Journal of Circuit Theory and Applications. — 2017. — Т. 45. — С. 989.

ПРИЛОЖЕНИЕ А

Основы теории решёток

В последние годы решётки служат алгоритмическим инструментом для решения широкого круга задач в информатике, математике и криптографии, особенно в квантовоустойчивых криптографических протоколах. Ниже приведены базовые понятия и известные алгоритмы, тесно связанные с нашей работой.

Базовые понятия

Пусть $\|\cdot\|$ — евклидова норма векторов из R^m. Векторы отмечаем полужирным (в переводе этого нет), матрицы пишем построчно; элементы матрицы M обозначаем $m_{i,j}$. Верхний индекс T означает транспонирование.

- **Решётка.** Пусть $b_1, \ldots, b_n \in R^m$ — линейно независимые столбцы, тогда множество всех линейных комбинаций их целочисленных коэффициентов — решётка, определяемая как

$$\Lambda(B) = \{\, Bx \mid x \in \mathbb{Z}^n \,\} = \{\, b = x_1 b_1 + \cdots + x_n b_n \mid x_1, \ldots, x_n \in \mathbb{Z} \,\}. \quad \text{(A.1)}$$

  где $B = [b_1, \ldots, b_n] \in \mathbb{R}^{m \times n}$ — матрица базиса, которая также может быть использована чтобы представлять решетку для простоты. $\{b_1, \ldots, b_n\}$ — группа базиса решетки. Размерность решётки n. Её детерминант $\det \Lambda = (\det B^T B)^{1/2}$, здесь B^T — транспонированная матрица B. При квадратной B имеем $\det \Lambda = \det B$. Детерминант представляет объём решётки в геометрическом представлении, обозначается как $\mathrm{vol}(\Lambda)$. Длина точки $b \in \mathbb{R}^m$ определяется как $\|b\| = (b^T b)^{1/2}$.
- **Последовательные минимумы.** Для n-мерной решётки Λ положительные числа $\lambda_1(\Lambda) \leq \lambda_2(\Lambda) \leq \cdots \leq \lambda_n(\Lambda)$ называются последовательными минимумами, где $\lambda_k(\Lambda)$ — наименьший радиус шара с центром в нуле, содержащего k линейно независимых векторов из Λ. Обозначим $\lambda_1 = \lambda_1(\Lambda)$ как длину кратчайшего ненулевого вектора из Λ.
- **Постоянная Эрмита.** Эрмитовым инвариантом решётки Λ называется

$$\gamma(\Lambda) = \frac{\lambda_1^2(\Lambda)}{\mathrm{vol}(\Lambda)^{2/n}} = \frac{\lambda_1^2(\Lambda)}{\det(\Lambda)^{2/n}}. \quad \text{(A.2)}$$

Постоянная Эрмита γ_n — максимальное значение $\gamma(\Lambda)$ для всех n-мерных решёток, или минимальное константное γ, удовлетворяющее $\lambda_1(\Lambda)^2 \leq \gamma(\det\Lambda)^{2/n}$ для всех решёток размености n соответственно.

- **QR-разложение.** У решёточной матрицы базиса B есть единственное разложение $B = QR \in \mathbb{R}^{m\times n}$, где $Q \in R^{m\times n}$, $R = [r_{i,j}]_{1\leq i,j\leq n} \in \mathbb{R}^{n\times n}$, здесь $Q \in \mathbb{R}^{m\times n}$ — изометрическая (столбцы ортогональны и единичной длины), а $R \in \mathbb{R}^{n\times n}$ — верхнетреугольная матрица с положительными диагональными элементами $r_{i,i}$. Коэффициенты Грама–Шмидта $\mu_{j,i} = r_{i,j}/r_{i,i}$ легко вычисляются из QR-разложения. Для целочисленной B коэффициенты $\mu_{j,i}$ обычно рациональны.
- **Задача кратчайшего вектора (SVP — shortest vector problem).** Дана группа базиса B решётки Λ.
  - Задача кратчайшего вектора (SVP): Требуется найти вектор $v \in \Lambda$, такой что $\|v\| = \lambda_1(\Lambda)$.
  - Приближённая задача кратчайшего вектора (α-SVP): Найти ненулевой вектор $v \in \Lambda$, удовлетворяющий $\|v\| \leq \alpha\,\lambda_1(\Lambda)$.
  - Эрмитова задача кратчайшего вектора (r-Hermite SVP): Найти ненулевой вектор $v \in \Lambda$, такой что $\|v\| \leq r\,\det(\Lambda)^{1/n}$.

  Параметр $\alpha \geq 1$ в α-SVP называется фактором аппроксимации. Обычно задача упрощается при увеличении α. При $\alpha = 1$ задачи α-SVP и SVP совпадают. Истинное значение λ_1 в α-SVP трудно вычислить из-за сложности SVP, поэтому решение α-SVP не всегда легко проверить. Задача r-Hermite SVP является вычислимой (относительно просто вычислимой), она определяется величиной $\det(\Lambda)^{1/n}$ вместо Λ_1. В результате, мы можем легко проверить решение, но не можем сравнить его с кратчайшим вектором.
- **Задача ближайшего вектора (CVP — closest vector problem)** Дана группа базиса B решётки Λ и целевой вектор $t \in \mathrm{span}(B)$.
  - Задача ближайшего вектора (CVP): Найти вектор $v \in \Lambda$, такой что расстояние $\|v - t\|$ может быть минимизировано, т.е. $\|v - t\| = \mathrm{dist}(\Lambda, t)$.
  - α-приближенная задача ближайшего вектора (α-CVP): Найти вектор $v \in \Lambda$, такой что расстояние $\|v - t\| \leq \alpha \times \mathrm{dist}(\Lambda, t)$
  - r-приближенная задача ближайшего вектора ((r-AbsCVP): Найти $v \in \Lambda$ такое, что $\|v - t\| \leq r$.

Определения аналогичны случаям для SVP; параметр $\alpha \geq 1$ в α-CVP играет ту же роль, что и в α-SVP. В r-AbsCVP параметр r может быть любым разумным значением, соизмеримым с $\mathrm{dist}(\Lambda, t)$, например $\det(\Lambda)^{1/n}$ в r-Hermite SVP.

Алгоритм LLL

Алгоритм 1 — Алгоритм LLL-редукции

Исходные параметры: базис решётки $\mathrm{b}_1, \ldots, \mathrm{b}_n \in \mathbb{Z}^m$, параметр δ
Результат: δ-LLL-редуцированный базис
Шаг 1: Орторгонализация Грама-Шмидта
Выполнить орторгонализацию Грама-Шмидта для базиса $\mathrm{b}_1, \ldots, \mathrm{b}_n$, обозначим результат как $\tilde{\mathrm{b}}_1, \ldots, \tilde{\mathrm{b}}_n \in \mathbb{R}^m$
Шаг 2: Редукция
цикл $i = 2$; $i < n$; $i = i + 1$ **выполнять**
  цикл $j = i - 1$; $i > 1$; $i = i - 1$ **выполнять**
    $c_{i,j} = \left\lfloor \dfrac{\langle \mathrm{b}_i, \tilde{\mathrm{b}}_j \rangle}{\langle \tilde{\mathrm{b}}_j, \tilde{\mathrm{b}}_j \rangle} \right\rceil$;
    $\mathrm{b}_i \leftarrow \mathrm{b}_i - c_{i,j} \cdot \mathrm{b}_j$;
  конец
конец
Шаг 3: Обмен
если $\exists\, i$ *такое, что* $\delta \cdot \|\tilde{b}_i\|^2 > \|\mu_{i+1,i}\tilde{b}_i + \tilde{b}_{i+1}\|^2$ **тогда**
  $\mathrm{b}_i \leftrightarrow \mathrm{b}_{i+1}$;
  перейти к шагу 1;
конец
Шаг 4: Вывод базиса $\mathrm{b}_1, \ldots, \mathrm{b}_n$

Алгоритм LLL — один из самых известных алгоритмов редукции решётки; он был предложен А. К. Ленстрой, Х. В. Ленстрой (мл.) и Л. Ловашем в 1982 г. [35]. Для n-мерной решётки этот алгоритм позволяет решать $\alpha = \left(\frac{2}{\sqrt{3}}\right)^n$ за полиномиальное время. Ниже приведены связанные понятия и алгоритмы.

– **LLL базис**: Базис $B = QR$ называется LLL-редуцированным или LLL-базисом при параметре редукции $\delta \in (1/4, 1]$, если выполняются условия:
  а) $\frac{|r_{i,j}|}{r_{i,i}} \leq \frac{1}{2}$, for all $j > i$;
  б) $\delta\, r_{i,i}^2 \leq r_{i,i+1}^2 + r_{i+1,i+1}^2$, for $i = 1, \ldots, n-1$.
  Очевидно, LLL-базис также удовлетворяет $r_{i,i}^2 \leq \alpha\, r_{i+1,i+1}^2$, для $\alpha = \frac{1}{(\delta - 1/4)}$.

Параметры, рассматриваемые в оригинальной литературе по алгоритму LLL, равны $\delta = \frac{3}{4}$, $\alpha = 2$. Известный результат о LLL-базисе показывает, что для любого $\delta < 1$ LLL-базис может быть получен за полиномиальное время и хорошо аппроксимирует последовательные минимумы

в) $\alpha^{-i+1} \leq \|b_i\|^2 \lambda_i^{-2} \leq \alpha^{n-1}, \quad \text{for } i = 1, \ldots, n;$

г) $\|b_1\|^2 \leq \alpha^{\frac{n-1}{2}} \left(\det \Lambda\right)^{2/n}.$

– **Алгоритм LLL**: Для заданного базиса $B = [b_1, \ldots, b_n] \in \mathbb{Z}^{m \times n}$ алгоритм может привести его к LLL-редуцированному виду или преобразовать в LLL-базис. Алгоритм состоит из трёх основных шагов: ортогонализация Грама–Шмидта, редукции и обмен. Конкретные шаги приведены в Алгоритме 1.

Алгоритм ближайшей плоскости Бабаи

Алгоритм ближайшей плоскости Бабаи [32] (далее — алгоритм Бабаи) применяется для решения CVP. Для n-мерной решётки он может получать фактор аппроксимации $\alpha = 2\left(\frac{2}{\sqrt{3}}\right)^n$ для α-CVP. Алгоритм состоит из двух этапов, первый из которых заключается в редукции решетки с помощью алгоритма LLL. Второй является процедурой уменьшения размера, который в основном вычисляет линейную комбинацию целочисленных коэффициентов, ближайших к целевому вектору t в LLL-базисе. Этот шаг по сути совпадает со вторым шагом в LLL-редукции. Подробные действия приведены в Алгоритме 2.

Алгоритм 2 — Алгоритм Бабаи

Исходные параметры: базис решётки $\mathrm{b}_1, \ldots, \mathrm{b}_n \in \mathbb{Z}^m$, параметр $\delta = 3/4$, целевой вектор $\mathrm{t} \in \mathbb{Z}^m$

Результат: вектор $\mathrm{x} \in \Lambda(B)$, такой что $\|\mathrm{x} - \mathrm{t}\| \leq 2^{n/2} \operatorname{dist}(\mathrm{t}, \Lambda(B))$

Шаг 1: LLL-редукция

Применить LLL-редукцию к базису B с параметром δ

Обозначим результат: $\tilde{\mathrm{b}}_1, \ldots, \tilde{\mathrm{b}}_n \in \mathbb{R}^m$

Шаг 2: Уменьшение размера

$\mathrm{b} \leftarrow \mathrm{t}$

цикл $j = n;\ j > 1\ j = j - 1$ **выполнять**

  $c_j = \left\lfloor \dfrac{\langle \mathrm{b}, \tilde{\mathrm{b}}_j \rangle}{\langle \tilde{\mathrm{b}}_j, \tilde{\mathrm{b}}_j \rangle} \right\rceil$

  $\mathrm{b} \leftarrow \mathrm{b} - c_j \cdot \mathrm{b}_j$

конец

Шаг 3: Вернуть $\mathrm{t} - \mathrm{b}$

ПРИЛОЖЕНИЕ Б

Алгоритм Шнорра для факторизации целых чисел

Метод решета Шнорра

Рассмотрим общую задачу факторизации, в которой задано целое число N, предполагается разложить его на два нетривиальных множителя $p < q$, так что $N = p \times q$. Метод решета для факторизации начинается с определения пары гладких отношений.

Пусть $p_i,\ i = 1, \ldots, n$ — первые n простых чисел вместе с p_0, удовлетворяющими $-1 = p_0 < 1 < p_1 < \cdots < p_n < p$. Множество $P = \{p_i\}_{i=0,\ldots,n}$ называется простым базисом. Число $p_0 = -1$ не является простым, однако включается для учёта знака целого числа. Целое число называется p_n-гладким, если все его простые делители меньше p_n; число p_n при этом называют пределом гладкости. Пара целых чисел (u_j, v_j) называется p_n-гладкой парой, если и u_j, и v_j являются p_n-гладкими. Более того, пара целых чисел (u_j, v_j) называется p_n-гладкой парой отношений (сокращённо sr-пара), если:

$$u_j \ = \ \prod_{i=1}^{n} p_i^{e_{i,j}}, \qquad u_j - v_j N \ = \ \prod_{i=0}^{n} p_i^{e'_{i,j}}, \tag{Б.3}$$

где $e_{i,j},\ e'_{i,j} \in N$, тогда имеем

$$\frac{u_j - v_j N}{u_j} \ \equiv \ \prod_{i=0}^{n} p_i^{e'_{i,j} - e_{i,j}} \ \equiv \ 1 \pmod{N}. \tag{Б.4}$$

Следует отметить, что гладкая пара отличается от sr-пары: sr-пара должна не только быть p_n-гладкой, но и удовлетворять более строгим условиям в уравнении Б.3. Пусть $S = \{(u_j, v_j)\}_{j=1,\ldots,n+1}$ — набор из $n+1$ sr-пар. Пусть существуют коэффициенты $t_1, \ldots, t_{n+1} \in \{0, 1\}$, такие что:

$$\sum_{j=1}^{n+1} t_j \left(e'_{i,j} - e_{i,j}\right) \ \equiv \ 0 \pmod{2}, \qquad i = 0, 1, \ldots, n. \tag{Б.5}$$

Обозначим $X \ = \ \prod_{i=0}^{n} p_i^{\frac{1}{2} \sum_{j=1}^{n+1} t_j \left(e'_{i,j} - e_{i,j}\right)}$, тогда

$$X^2 - 1 \ = \ (X+1)(X-1) \ \equiv \ 0 \pmod{N}. \tag{Б.6}$$

Если $X \not\equiv \pm 1 \pmod N$, то нетривиальный фактор числа N получается как $\gcd(X \pm 1,\, N)$.

Поскольку размерность системы линейных уравнений равна $O(n)$ и она решается за $O(n^3)$ операций, эту малую часть вычислений при факторизации N мы опускаем. Следовательно, задача факторизации сводится к задаче поиска sr-пары. В дальнейшем эта задача будет преобразована в задачу ближайшего вектора на решётке.

Построение решётки и целевого вектора

sr-пары будут получены из приближённого решения задачи CVP в алгоритме Шнорра. Сначала опишем построение простой решётки $\Lambda(B_{n,c})$ и целевого вектора $t \in \mathbb{R}^{n+1}$; здесь $c > 0$ — настраиваемый параметр. Матрица решётки $B_{n,c} = [b_1, \ldots, b_n] \in \mathbb{R}^{(n+1)\times n}$ задаётся

$$B_{n,c} = \begin{pmatrix} f(1) & 0 & \ldots & 0 \\ 0 & f(2) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & f(n) \\ N^c \ln p_1 & N^c \ln p_2 & \ldots & N^c \ln p_n \end{pmatrix}, \qquad t = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ N^c \ln N \end{pmatrix}. \tag{Б.7}$$

где функции $f(i)$ при $i = 1, \ldots, n$ — случайные перестановки диагональных элементов $(\sqrt{\ln p_1}, \sqrt{\ln p_2}, \ldots, \sqrt{\ln p_n})$.

Точку решётки или вектор можно представить целочисленной комбинацией базиса решетки: $b = \sum_{i=1}^{n} e_i b_i \in \Lambda(B_{n,c})$, причём $e_i \in Z$. Далее будем полагать, что (u, v) — p_n-гладкая пара и $\gcd(u, v) = 1$. Тогда u, v выражаются через произведение простых чисел из простого базиса:

$$u \;=\; \prod_{e_i>0} p_i^{e_i}, \qquad v \;=\; \prod_{e_i<0} p_i^{-e_i}. \tag{Б.8}$$

В таком представлении гладкой паре (u, v) взаимно однозначно соответствует вектор $b = (e_1, \ldots, e_n)$ на решётке, пишем $b \sim (u, v)$. Таким образом, вектор решётки кодирует гладкую пару.

Задача ближайшего вектора (CVP) формулируется как поиск вектора $b_0 \in \Lambda(B_{n,c})$, минимально удалённого от t:

$$b_0 \;=\; \arg\min_{b\in\Lambda} \|b-t\|. \tag{Б.9}$$

Согласно приведённому выше определению справедливо отношение:

$$\|b-t\|^2 \;\geq\; \ln(uv) \;+\; N^{2c}\left|\ln\tfrac{u}{vN}\right|^2. \tag{Б.10}$$

Уравнение выполняется тогда и только тогда, когда $e_i \in \{-1, 0, 1\}$, то есть u, v не содержат квадратных множителей. Константа N^{2c} выступает «весом», управляемым параметром c. При $N^{2c} \gg \ln(uv)$ главной частью равенства становится $N^{2c}\,|\ln\frac{u}{vN}|^2$. Следовательно, параметр c (также называемый параметром точности) влияет на величину $|\ln\frac{u}{vN}|^2$, а значит и на $|u - vN|$. Из неравенства Б.10 видно: чем короче вектор расстояния $b-t$, тем меньше $|u-vN|$ и тем выше вероятность того, что (u, v) является sr-парой. Дополнительное обсуждение этой зависимости приведено в следующем разделе материала.

Решение CVP

Существует два хорошо изученных подхода к решению задачи ближайшего вектора (CVP) или её аппроксимации. Первый основан на методе решета, впервые предложенном Айтаем с соавторами в 2001 г. [36]. Второй базируется на алгоритме Бабаи: сначала выполняют редукцию решётки (например, алгоритмом LLL), чтобы получить относительно короткий базис, а затем применяют процедуру уменьшения размера для получения приближённого решения CVP. Шнорр использовал именно второй подход. Фактически для повышения эффективности алгоритма привлекают более совершенные методы редукции, такие как BKZ [37], HKZ, ENUM [37—40] и др. Однако эти методы слишком сложны и требуют специальных знаний, выходящих за рамки данной работы. Поэтому далее (также и в основном тексте) под алгоритмом Бабаи мы будем подразумевать реализацию с использованием LLL-редукции, которая проста и относительно легко понимается. При этом принцип квантового ускорения алгоритма Бабаи остается общим для любого метода редукции решётки.

ПРИЛОЖЕНИЕ В

Сублинейная схема о размерности решетки

Исторические результаты

В этом разделе мы обсуждаем выбор размерности решётки n в алгоритме Шнорра. Размерность определяется мощностью простого базиса и существенно влияет на эффективность алгоритма. С одной стороны, при большом n число гладких пар на простом базисе резко возрастает, упрощая их поиск. С другой — время работы редукции базиса и решения систем линейных уравнений растёт вместе с n. Следовательно, необходимо сбалансировать оба фактора. В оригинальных работах Шнорра [29; 30; 41] вопрос выбора n раскрыт не полностью, поэтому в разных публикациях встречаются разные подходы. В обновлённой версии Шнорра 2021 г. [30] для конкретных примеров используется сублинейная размерность, но без объяснений. Например, при факторизации 400-битного числа размерность решетки составляет 48, что близко к схеме $400/\log_2 400 \approx 46$. Во многих других работах размерность решётки n обычно предполагают полиномиальной от длины m большого целого числа N. Объяснение дано из ограничения на предел гладкости p_n. В решете Шнорра предел гладкости обычно принимают удовлетворяющим условию:

$$p_n \;\approx\; (\log N)^{\alpha} \;=\; m^{\alpha}, \qquad \alpha > 0. \tag{B.11}$$

Согласно теореме о простых числах получаем

$$n \;\approx\; \frac{(\log N)^{\alpha}}{\alpha \log\log N} \;=\; \frac{m^{\alpha}}{\alpha \log m}. \tag{B.12}$$

При $\alpha = 1$ имеем

$$n \;=\; \frac{m}{\log m}, \tag{B.13}$$

что даёт сублинейную размерность по длине числа N. При $\alpha > 1$ n становится полиномиальным по m. Таким образом, именно выбор α определяет размерность решётки.

Параметр α связан с математической зависимостью между коротким вектором и гладкой парой. Условие, при котором короткие векторы дают гладкие пары, сформулировано Шнорром в следующей лемме.

Лемма 1. Если $\|b - t\|^2 = O(\log N)$ и $v \leq N^{c-1} p_n \big(n/\log N\big)^{1/2}$, то, с высокой вероятностью, $|u - vN| = O(p_n)$.

Здесь c — параметр точности. Лемма утверждает, что когда квадрат нормы короткого вектора равен $O(\log N)$, то скорее ввсего sr-пары могут быть получены. Мы принимаем $O(\log N)$ как теоретическую границу длины короткого вектора.

Возникает следующий важный вопрос: существуют ли короткие векторы, удовлетворяющие этому условию, и достаточно ли их. Шнорр показал, что их много при $\alpha > 2$. Величина α пропорциональна пределу гладкости по формуле B.11. В методе решета чем больше гладкая граница p_n, тем легче найти гладкие пары, но тем больше их требуется всего. Шнорр доказал, что при $\alpha > (2c - 1)/(c - 1) > 2$ существует большое количество коротких векторов, формирующих гладкие пары, что ведёт к полиномиальной размерности схемы.

Ниже мы рассматриваем связь между коротким вектором и гладкой парой с точки зрения существования короткого вектора. Сначала приведена линейная схема размерности n, основанная на первой теореме Минковского [42]. Затем, опираясь на предположение о плотности в алгоритме Шнорра [30], выводится сублинейная схема размерности решётки.

Линейная схема

Проблема существования заключается в том, существует ли вектор $b \in \Lambda(B_{n,c})$ такой, что выполняется условие $\|b - t\|^2 = O(\log N)$. Мы оцениваем расстояние от целевого вектора t до решётки Λ через длину λ_1 кратчайшего вектора в расширенной решётке $\bar{B}_{n,c} = [B_{n,c},\, t]$. Поскольку детерминант $\bar{B}_{n,c}$ известен, верхнюю границу для λ_1 можно получить по первой теореме Минковского, формулируемой так.

Лемма 2 (первая теорема Минковского). Для любой полной решётки Λ размерности n

$$\lambda_1(\Lambda)^2 \;\leq\; n\,(\det \Lambda)^{2/n}. \tag{B.14}$$

Первая теорема Минковского даёт верхнюю границу для кратчайшего ненулевого вектора, то есть первой последовательной минимальной λ_1. На основе этой оценки получаем следующий результат.

Утверждение 1. Если размерность решётки $B_{n,c}$ равна $n = \log N$, то существует вектор $b \in \Lambda(\bar{B}_{n,c})$, для которого

$$\|b - t\|^2 = O(\log N). \tag{B.15}$$

Доказательство. Обозначим длину кратчайшего вектора расширенной решётки $\bar{B}_{n,c}$ через λ_1. Здесь мы используем порядок λ_1 чтобы оценить $\mathrm{dist}(B_{n,c}, t)$ между решеткой и целевым вектором, полагая что $\mathrm{dist}(B_{n,c}, t) = O(\lambda_1)$. По первой теореме Минковского имеем:

$$\lambda_1^2 \;\le\; (n+1)\left(\det \bar{B}_{n,c}\right)^{2/(n+1)}. \tag{B.16}$$

С учётом конструкции решётки получаем

$$\left(\det \bar{B}_{n,c}\right)^{2/(n+1)} \;=\; \Big(\prod_{i=1}^{n} f(i)\Big)^{2/(n+1)} \left(N^c \log N\right)^{2/(n+1)}. \tag{B.17}$$

Предположим, что диагональные элементы выбираются из множества $\{1, 2\}$, причём двойки занимают долю $(n+1)/(3n)$, — это обеспечивает достаточное количество различных перестановок для случайных решёток. Тогда

$$\Big(\prod_{i=1}^{n} f(i)\Big)^{2/(n+1)} = \left(2^{(n+1)/3}\right)^{2/(n+1)} = 2^{2/3} = O(1). \tag{B.18}$$

Подставляя в уравнение B.18 и $n = \log N$ в B.17, получаем

$$\left(\det \bar{B}_{n,c}\right)^{2/(n+1)} = O\big(N^{\,2c/(n+1)}\big) = O\big(2^{\,2cn/(n+1)}\big) = O(1). \tag{B.19}$$

Отсюда следует

$$\lambda_1^2 \;\le\; n\,O(1) \;=\; O(\log N), \tag{B.20}$$

что и завершает доказательство.

Отметим, что в построении решётки диагональные элементы берутся из $\{1, 2\}$, а число двоек приблизительно равно $(n+1)/(3n)$. Это условие можно обобщить до

$$\prod_{i=1}^{n} f(i)^{2/(n+1)} \ \sim \ O(1). \tag{B.21}$$

В первой теореме Минковского верхнюю границу можно уточнить, используя постоянные Эрмита. Рассмотрим соотношение:

$$\gamma \ = \ \frac{\lambda_1^2(\Lambda)}{(\det \Lambda)^{2/n}}. \tag{B.22}$$

Определение 1. Обозначим через γ_n наибольшее значение, удовлетворяющее B.22 для всех n-мерных решёток. Тогда γ_n называют *постоянной Эрмита* размерности n.

На самом деле γ_n является точной верхней границей: для каждого $n > 1$ существует решётка, в которой достигается равенство $\gamma_n = \lambda_1^2(\Lambda)/(\det \Lambda)^{2/n}$. Такие решётки называют критическими. Но вычисление точного значения γ_n обычно сложно, что также является основной проблемой в изучении геометрических чисел Минковского. Точные значения γ_n известны лишь для $1 \leq n \leq 8$ и $n = 24$; асимптотически лучшая оценка [43]

$$\lambda_1^2 \ \leq \ \gamma_n \, (\det \Lambda)^{2/n} \ \leq \ \frac{1.744\, n}{2e\pi} \, (\det \Lambda)^{2/n}. \tag{B.23}$$

Используя B.23 для оценки λ_1, получаем тот же вывод, что и в утверждение 1.

Сублинейная схема

Поскольку первая теорема Минковского даёт лишь верхнюю границу для длины кратчайшего вектора, у многих случайных решёток действительная длина этого вектора значительно отличается от оценки. Этот разрыв удобно измерять относительной плотностью решётки $\mathrm{rd}(\Lambda)$. Относительная плотность $\mathrm{rd}(\Lambda)$ определяется как отношение действительной длины кратчайшего вектора λ_1 к верхней границе, полученной через постоянную Эрмита. Из уравнения B.23 сле-

дует, что $0 < \mathrm{rd}(\Lambda) \leq 1$. Точное определение:

$$\mathrm{rd}(\Lambda) = \frac{\lambda_1}{\sqrt{\gamma_n}\,(\det \Lambda)^{1/n}}. \tag{B.24}$$

Когда относительная плотность близка к 1, это означает, что оптимальные векторы базиса решётки имеют одинаковую длину, а точки решётки расположены плотно.

Шнорр сделал следующее допущение о относительной плотности решёток, используемых для поиска гладких пар, анализируя эффективность алгоритма.

Допущение 1. Случайная решётка Λ с базисом $B = [b_1, \ldots, b_n]$ имеет относительную плотность, удовлетворяющую

$$\mathrm{rd}(\Lambda) \;\leq\; \left(\sqrt{\frac{e\pi}{2n}}\,\frac{\lambda_1}{\|b_1\|}\right)^{1/2}, \tag{B.25}$$

то есть b_1 и $\mathrm{rd}(\Lambda)$ достаточно малы. Поскольку $\lambda_1/\|b_1\| \leq 1$, из этого допущения следует

$$\mathrm{rd}(\Lambda) = \frac{\lambda_1}{\sqrt{\gamma_n}\,(\det \Lambda)^{1/n}} \;\leq\; \left(\frac{e\pi}{2n}\right)^{1/4}. \tag{B.26}$$

Следовательно, имеем следующие результаты.

Утверждение 2. Если размерность решётки $B_{n,c}$ удовлетворяет $n = \frac{2c \log N}{\log\log N}$, а относительная плотность удовлетворяет Допущению 1, то существует вектор $b \in \Lambda(B_{n,c})$, такой что

$$\|b - t\|^2 = O(\log N). \tag{B.27}$$

Доказательство. Из равенства B.26 имеем

$$\lambda_1^2 \;\leq\; \left(\frac{e\pi}{2n}\right)^{1/2} \gamma_n\,(\det \Lambda)^{2/n}. \tag{B.28}$$

Подставляя B.17 и B.18 в уравнение выше получаем

$$\lambda_1^2 \;\leq\; \left(\frac{e\pi}{2n}\right)^{1/2} \gamma_n\, N^{2c/n}. \tag{B.29}$$

Если принять $n = 2c \log N / \log\log N$, то

$$\lambda_1^2 \;\leq\; \left(\frac{e\pi}{2n}\right)^{1/2} \frac{1.744}{2e\pi} \sqrt{\frac{2c \log N}{\log\log N}}\, \log N = O(\log N). \tag{B.30}$$

Здесь величина $\sqrt{2c \log N / \log\log N}$ — порядок меньший, чем $\log N$, поэтому в итоговом выражении она опущена. Что и требовалось доказать.

Игнорирование указанной низшего порядка величины оправдано. При $c = 1$ и $N \approx 2^{1024}$ получаем

$$\left(\frac{e\pi}{2}\right)^{1/2} \frac{1.744}{2e\pi} \sqrt{\frac{2 \log N}{\log\log N}} \;\approx\; 3.0960 \sim O(1). \tag{B.31}$$

А при $N \approx 2^{2048}$ имеем

$$\left(\frac{e\pi}{2}\right)^{1/2} \frac{1.744}{2e\pi} \sqrt{\frac{2 \log N}{\log\log N}} \;\approx\; 4.1641 \sim O(1). \tag{B.32}$$

Следовательно, при выполнении Допущения 1 выбор размерности $n = 2c \log N / \log\log N$ рационален, и квадрат нормы кратчайшего вектора гарантированно остаётся порядка $O(\log N)$. Это означает, что с высокой вероятностью гладкая пара может быть получена из ближайшего вектора решётки, как утверждается в Лемме 1.

ПРИЛОЖЕНИЕ Г

Предварительная обработка: детали факторизации

Построение решётки и целевого вектора

Чтобы показать вычислительные шаги до квантовой части в качестве примера возьмём факторизацию числа $N = 48567227$ на 5 кубитах. К ним относятся построение решётки и целевого вектора, LLL-редукция и выполнение алгоритма ближайшей плоскости Бабаи. Случаи с 3 и 10 кубитами также будут приведены. Здесь применяется сублинейная схема размерности решётки. Размерность, необходимая для факторизации $N = 48567227$, равна $\log N / \log\log N = 26/5 \approx 5$. База простых чисел состоит из первых пяти простых: $\{-1, 2, 3, 5, 7, 11\}$.

Чтобы получить достаточно случайных целочисленных решёток, мы слегка изменяем построение решётки из уравнения Б.7. Во-первых, вместо диагонали $\sqrt{\ln p_i}$ используем $\lceil i/2 \rfloor$, где $\lceil \rfloor$ — округление к ближайшему целому. Во-вторых, чтобы получить различные fac-отношения, применяем случайную перестановку f диагональных элементов. Кроме того, вместо «весового» элемента N^c используем 10^c. Таким образом, если c — целое число, решётку легко сделать целочисленной, а c прямо задаёт точность. Конкретная структура решётки задаётся уравнениями Г.33 и Г.34:

$$B_{n,c} = \begin{pmatrix} f(1) & 0 & \dots & 0 \\ 0 & f(2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & f(n) \\ \lfloor 10^c \ln 2 \rceil & \lfloor 10^c \ln 3 \rceil & \dots & \lfloor 10^c \ln 11 \rceil \end{pmatrix}, \tag{Г.33}$$

$$t_n = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \lfloor 10^c \ln N \rceil \end{pmatrix}. \tag{Г.34}$$

Здесь $B_{n,c}$ — матричная форма решётки, каждый столбец является вектором базиса. Подстрочные индексы обозначают размерность n и параметр точности c. В случае с 5 кубитами размерность равна 5, а параметр точности — 4. Элементы $f(i)$ на диагонали — случайная перестановка множества $\{\lceil 1/2 \rfloor, \ldots, \lceil 5/2 \rfloor\} = \{1, 1, 2, 2, 3\}$. Соответственно решётка и целевой вектор, относящиеся к sr-паре, даны в уравнениях Г.35 и Г.36:

$$B_{5,4} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 6931 & 10986 & 16094 & 19459 & 23979 \end{pmatrix}, \tag{Г.35}$$

$$t_5 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 176985 \end{pmatrix}. \tag{Г.36}$$

Аналогично для случая с 3 кубитами выполняются условия $n = 3$ и $c = 1.5$. Соответствующие решётка и целевой вектор, соответсвующий sr-паре:

$$B_{3,1.5} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \\ 22 & 35 & 51 \end{pmatrix}, \qquad t_3 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 240 \end{pmatrix}. \tag{Г.37}$$

В случае с 10 кубитами выполняются условия $n = 10$ и $c = 4$. Решётка и целевой вектор для sr-пары приведены в уравнениях Г.38 (пропущено в изначальной работе) и Г.39:

$$B_{10,4} = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 6931 & 10986 & 16094 & 19459 & 23979 & 25649 & 28332 & 29444 & 31355 & 33673 \end{pmatrix} \tag{Г.38}$$

$$t_{10} = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 331993)^T. \tag{Г.39}$$

Решение CVP с помощью алгоритма Бабаи

Гладкую пару отношений можно получить, решив CVP на построенной выше решётке. До применения квантового метода приближённое оптимальное решение CVP извлекают классическим алгоритмом редукции решётки (алгоритм Бабаи). Сначала на базис решётки выполняется LLL-редукция с параметром $\delta = 3/4$. LLL-редуцированные базисы для трёх случаев факторизации обозначаются, соответственно, $D_{3,1.5}$ (Г.40), $D_{5,4}$ (Г.41) и $D_{10,4}$ (Г.42):

$$D_{3,1.5} = \begin{pmatrix} 1 & -4 & -3 \\ -2 & 1 & 2 \\ 2 & 2 & 0 \\ 3 & -2 & 4 \end{pmatrix} \tag{Г.40}$$

$$D_{5,4} = \begin{pmatrix} 6 & -8 & 2 & -4 & -4 \\ -4 & -3 & 11 & -5 & -3 \\ 6 & 6 & 3 & 0 & -3 \\ 4 & -2 & 0 & 12 & 4 \\ -2 & 2 & -6 & -2 & 1 \\ -3 & 5 & -3 & 4 & -17 \end{pmatrix} \tag{Г.41}$$

$$D_{10,4} = \begin{pmatrix} 0 & 0 & 3 & 0 & 0 & 0 & 3 & 0 & -3 & -3 \\ 0 & 2 & 0 & 4 & -4 & 0 & 4 & -2 & 4 & 0 \\ -3 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 4 & -2 & -2 & 0 & -1 & 0 & 0 \\ 2 & 0 & 0 & -2 & 0 & 1 & -1 & 0 & 4 & 0 \\ 0 & 0 & -3 & -3 & 0 & 0 & 1 & 0 & -3 & 3 \\ -3 & 3 & 1 & 0 & 1 & 2 & 1 & 2 & -2 & -1 \\ 0 & -2 & 0 & 1 & 2 & -1 & 1 & -3 & 3 & -3 \\ 2 & -2 & 0 & -2 & 0 & 1 & 2 & 0 & 2 & 2 \\ 2 & -2 & 2 & 0 & 2 & -2 & 2 & 2 & 0 & 0 \\ 0 & -2 & -2 & 0 & 1 & 3 & 1 & -2 & -2 & -1 \end{pmatrix} \tag{Г.42}$$

Затем выполняется процедура уменьшения размера. Берётся самый «длинный» вектор LLL-базиса (правый столбец матрицы $D_{5,4}$) и пошагово вычитается из целевого вектора t_5, используя округлённые коэффициенты Грама–Шмидта μ_i $(i = 1, \ldots, 5)$, пока не будет обработан самый «короткий» базисный вектор (левый столбец). В конце получают вектор расстояния $\bar{t}_5$ и приближённый ближайший вектор b_{op}. Поскольку длина $\bar{t}_5$ характеризует качество решения CVP, её также называют «коротким вектором» в контексте CVP. Классические оптимальные решения, найденные алгоритмом Бабаи, приведены в Г.43–Г.46.

$$b_{\text{op}} = (2\ 4\ 9\ 8\ 0\ 176993)^T, \qquad \bar{t}_5 = b_{\text{op}} - t_5 = (2\ 4\ 9\ 8\ 0\ 8)^T. \tag{Г.43}$$

$$b_{\text{op}} = (0\ 4\ 4\ 242)^T, \qquad \bar{t}_3 = b_{\text{op}} - t_3 = (0\ 4\ 4\ 2)^T. \tag{Г.44}$$

$$b_{\text{op}} = (3\ 4\ 0\ 1\ 2\ 3\ 2\ 3\ 2\ 2\ 331993)^T, \tag{Г.45}$$

$$\bar{t}_{10} = b_{\text{op}} - t_{10} = (3\ 4\ 0\ 1\ 2\ 3\ 2\ 3\ 2\ 2\ 0)^T. \tag{Г.46}$$

Приближённый ближайший вектор оказывается довольно далёк от целевого t_5: $\|\bar{t}_5\|^2 = 229$. Во всех трёх случаях факторизации можно получить вектор, более близкий (то есть более короткий), чем у алгоритма Бабаи, посредством квантовой оптимизации.

Гамильтониан задачи

В основном тексте было описано, как построить гамильтониан задачи, отображая двоичные переменные $x_i,\ i = 1, \ldots, n$ на элементы Паули-Z,

$$H_c = \left\| t - \sum_{i=1}^{n} \hat{x}_i d_i - b_{\text{op}} \right\|^2 = \sum_{j=1}^{n+1} \left| t_j - \sum_{i=1}^{n} \hat{x}_i d_{i,j} - b_{\text{op}}^j \right|^2 \tag{Г.47}$$

Мы кодируем плавающие переменные $x_i \in \{-1, 0, 1\}$ (определённые промежуточными вычислениями алгоритма Бабаи) одним кубитом. Квантовый оператор $\hat{x}_i$ отображается в базис Паули-Z по следующим правилам:

$$\hat{x}_i = \begin{cases} \dfrac{I - \sigma_z^i}{2}, & \text{если } c_i \leq \mu_i, \\ \dfrac{\sigma_z^i - I}{2}, & \text{если } c_i > \mu_i. \end{cases} \tag{Г.48}$$

Если коэффициент был округлён вниз, т.е. $c_i \leq \mu_i$, то его значение увеличивается на 1 или остаётся неизменным; в этом случае плавающее значение $x_i \in \{0, 1\}$ соответствует собственным значениям оператора $\dfrac{I - \sigma_z^{(i)}}{2}$ и наоборот. Тем самым информация об округлении c_i в алгоритме Бабаи определяет кодирование x_i. Нетрудно видеть, что более низкое энергетическое состояние гамильтониана даёт приближённое решение ближайшего вектора в решётке Λ, поскольку гамильтониан соответствует функции потерь.

Для случая с пятью кубитами гамильтониан можно записать как $H_{c5} = \sum_{j=1}^{6} \hat{h}_j$, где

$$\begin{cases} \hat{h}_1 = (\, 6\hat{x}_1 - 8\hat{x}_2 + 2\hat{x}_3 - 4\hat{x}_4 - 4\hat{x}_5 + 2\,)^2, \\ \hat{h}_2 = (\, -4\hat{x}_1 - 3\hat{x}_2 + 11\hat{x}_3 - 5\hat{x}_4 - 3\hat{x}_5 + 4\,)^2, \\ \hat{h}_3 = (\, 6\hat{x}_1 + 6\hat{x}_2 + 3\hat{x}_3 - 0\hat{x}_4 - 3\hat{x}_5 + 9\,)^2, \\ \hat{h}_4 = (\, 4\hat{x}_1 - 2\hat{x}_2 + 0\hat{x}_3 + 12\hat{x}_4 + 4\hat{x}_5 + 8\,)^2, \\ \hat{h}_5 = (\, -2\hat{x}_1 + 2\hat{x}_2 - 6\hat{x}_3 - 2\hat{x}_4 + \hat{x}_5\,)^2, \\ \hat{h}_6 = (\, -3\hat{x}_1 + 5\hat{x}_2 - 3\hat{x}_3 + 4\hat{x}_4 - 17\hat{x}_5 + 8\,)^2. \end{cases} \tag{Г.49}$$

Конкретный способ кодирования каждой переменной $x_i,\ i = 1, \ldots, 5$ определяется по промежуточным данным алгоритма Бабаи и приведён в таблице Г.1.

Таблица Г.1 – Кодирование кубитов для 5-кубитного случая. Подскрипт «j» убывает слева направо.

шаги	1 (x_5)	2 (x_4)	3 (x_3)	4 (x_2)	5 (x_1)
μ_j	-8731.5607	3882.5019	-1837.4760	-354.467	-3092.4957
c_j	-8732	3883	-1837	-354	-3092
$\mu_j - c_j$	0.4393	-0.4981	-0.4760	-0.4669	-0.4957
кодирование	$(0, 1)$	$(0, -1)$	$(0, -1)$	$(0, -1)$	$(0, -1)$

Соответственно, 5-кубитный гамильтониан сводится к уравнению Г.50.

$$\begin{aligned} H_{c5} = {} & 781\, I - 14\sigma_z^1 - 640\sigma_z^2 - 810\sigma_z^3 - 213\sigma_z^4 - 4.5\sigma_z^5 - 13.5\sigma_z^1\sigma_z^2 + 3.5\sigma_z^1\sigma_z^3 \\ & + 18\sigma_z^1\sigma_z^4 + 17.5\sigma_z^1\sigma_z^5 - 29\sigma_z^2\sigma_z^3 + 19.5\sigma_z^2\sigma_z^4 - 34\sigma_z^2\sigma_z^5 - 31.5\sigma_z^3\sigma_z^4 \\ & - 2.5\sigma_z^3\sigma_z^5 + 4.5\sigma_z^4\sigma_z^5. \end{aligned} \tag{Г.50}$$

Кодирование кубитов и гамильтониан для 3-кубитного случая даны в таблице Г.2 и уравнении Г.51.

Таблица Г.2 – Кодирование кубитов для 3-кубитного случая. Подскрипт «j» убывает слева направо.

шаги	1 (x_3)	2 (x_2)	3 (x_1)
μ_j	33.5812	-20.4974	21.6667
c_j	34	-20	22
$\mu_j - c_j$	-0.4188	-0.4974	-0.3333
кодирование	$(0, -1)$	$(0, -1)$	$(0, -1)$

$$H_{c3} = 43.5\, I - 4\, \sigma_z^1\sigma_z^2 + 2.5\, \sigma_z^1\sigma_z^3 - 1.5\, \sigma_z^1 + 3\, \sigma_z^2\sigma_z^3 - 3.5\, \sigma_z^2 - 4\, \sigma_z^3. \tag{Г.51}$$

Кодирование кубитов и гамильтониан для 10-кубитного случая даны в таблице Г.3 и уравнении Г.52.

Таблица Г.3 – Кодирование кубитов для 10-кубитного случая. Подскрипт «j» убывает слева направо.

шаги	1 (x_{10})	2 (x_9)	3 (x_8)	4 (x_7)	5 (x_6)	6 (x_5)	7 (x_4)	8 (x_3)	9 (x_2)	10 (x_1)
μ_j	21514.149	-45688.541	-29225.450	-5953.325	29891.446	23868.721	42395.337	-18221.276	-29823.805	5952.889
c_j	21514	-45689	-29225	-5953	29891	23869	42395	-18221	-29824	5953
$\mu_j - c_j$	0.149	0.459	-0.450	-0.325	0.446	-0.279	0.337	-0.276	0.195	-0.111
кодирование	$(0,1)$	$(0,1)$	$(0,-1)$	$(0,-1)$	$(0,1)$	$(0,-1)$	$(0,1)$	$(0,-1)$	$(0,1)$	$(0,-1)$

$$
\begin{aligned}
H_{c10} = \frac{1}{4}\Big(&708\,I + 22\sigma_z^1\sigma_z^2 + 16\sigma_z^1\sigma_z^3 + 8\sigma_z^1\sigma_z^4 - 14\sigma_z^1\sigma_z^5 + 8\sigma_z^1\sigma_z^6 + 4\sigma_z^1\sigma_z^7 - 8\sigma_z^1\sigma_z^8 \\
&- 10\sigma_z^1\sigma_z^9 - 22\sigma_z^1\sigma_z^{10} - 46\sigma_z^1 - 14\sigma_z^2\sigma_z^3 + 20\sigma_z^2\sigma_z^4 + 14\sigma_z^2\sigma_z^5 - 12\sigma_z^2\sigma_z^6 \\
&+ 20\sigma_z^2\sigma_z^7 - 24\sigma_z^2\sigma_z^8 - 28\sigma_z^2\sigma_z^9 + 2\sigma_z^2\sigma_z^{10} - 16\sigma_z^2 - 18\sigma_z^3\sigma_z^4 + 10\sigma_z^3\sigma_z^5 \\
&+ 36\sigma_z^3\sigma_z^6 + 16\sigma_z^3\sigma_z^8 + 6\sigma_z^3\sigma_z^9 - 30\sigma_z^3\sigma_z^{10} - 78\sigma_z^3 + 28\sigma_z^4\sigma_z^5 - 26\sigma_z^4\sigma_z^6 \\
&+ 10\sigma_z^4\sigma_z^8 + 16\sigma_z^4\sigma_z^9 - 4\sigma_z^4\sigma_z^{10} - 72\sigma_z^4 + 10\sigma_z^5\sigma_z^6 + 24\sigma_z^5\sigma_z^7 + 20\sigma_z^5\sigma_z^8 \\
&+ 12\sigma_z^5\sigma_z^9 - 8\sigma_z^5\sigma_z^{10} - 116\sigma_z^5 - 5\sigma_z^6\sigma_z^7 + 22\sigma_z^6\sigma_z^8 - 6\sigma_z^6\sigma_z^9 - 36\sigma_z^6\sigma_z^{10} \\
&- 120\sigma_z^6 - 16\sigma_z^7\sigma_z^8 + 16\sigma_z^7\sigma_z^9 + 20\sigma_z^7\sigma_z^{10} - 84\sigma_z^7 + 34\sigma_z^8\sigma_z^9 \\
&- 42\sigma_z^8\sigma_z^{10} - 36\sigma_z^8 + 18\sigma_z^9\sigma_z^{10} - 74\sigma_z^9 - 24\sigma_z^{10}\Big).
\end{aligned}
\tag{Г.52}
$$

Энергетический спектр и целевое состояние

Численно просматриваем энергетический спектр гамильтониана задачи. Ниже приводим лишь десять низших уровней энергии и соответствующие квантовые состояния; при наличии — также соответствующие sr-пары. Следует различать предел гладкости B_2 для $|u - vN|$ и предел B_1 для самой пары (u, v). В 5-кубитном случае берём $B_2 = p_{50} = 229$, а размерность соответствующей системы линейных уравнений (обозначим eq-dim) равна 51. Данные для 3- и 10-кубитного случаев даны в таблице Г.4. Размерность системы $\sim 2n^2$ — полиномиальна от n, поэтому ослабить предел B_2 разумно. С одной стороны, при методе решета Шнорра размерность базы простых (то есть размерность решётки) мала, и решение системы требует небольших ресурсов. С другой стороны, алгоритм предъявляет высокие требования к качеству коротких векторов, что резко увеличивает совокупную трудоёмкость. Умеренное ослабление B_2 снижает требования к вектору, увеличивая число уравнений, но в целом повышая эффективность алгоритма.

Таблица Г.4 – Два предела гладкости для трёх случаев факторизации.

случай	**B1-dim**	**B1**	**B2-dim**	**B2**	**eq-dim**
3 кубита	3	5	15	47	16
5 кубит	5	11	50	229	51
10 кубит	10	29	200	1223	201

В таблице Г.5 первый столбец — десять низших уровней энергии гамильтониана из уравнения Г.50; второй — сами энергии, т. е. квадраты норм коротких векторов. Третий столбец содержит собственные состояния. Строка 1 — основное состояние, его энергия — 186. Длина соответствующего вектора минимальна, однако sr-пара не получена: связь между коротким вектором и sr-парой носят вероятностный характер. Энергия четвёртого возбуждённого состояния равна 215, и $|u - vN| = 12097706 = 2{\cdot}41{\cdot}43{\cdot}47{\cdot}73$ гладко относительно B_2; из состояния (00111) получается sr-пара. Седьмое возбуждённое состояние (00000) даёт энергию 229 — это оптимум алгоритма Бабаи. Таким образом, квантовый метод выдаёт более короткий вектор и sr-пару, что делает состояние (00111) целевым.

Таблица Г.5 – Первые десять наименьших уровней энергии и соответствующие квантовые состояния. Четвёртое возбуждённое состояние порождает гладкую пару отношений, и соответствующее значение $|u - vN| = 12097706 = 2 \cdot 41 \cdot 43 \cdot 47 \cdot 73$ является гладким относительно границы B_2, что делает это состояние целевым для 5-кубитного случая.

уровень	**энергия**	**состояние**	u	v	$\lvert u - vN \rvert$	гладкое
0	186	0 1 0 1 1 0	21435888100	441	89·199337	нет
1	189	0 1 1 1 0 0	3401399712	7	53·3191	нет
2	193	1 1 1 0 0 0	1215290846	25	3^2·370057	нет
3	198	1 0 0 0 1 1	776562633	16	512999	нет
4	215	0 0 1 1 1 1	11789738455	243	2^4·43·47·73	да
5	218	1 1 0 0 0 0	243045684	5	205949	нет
6	222	1 1 1 1 0 0	4167418.11	8575	249693139	нет
7	229	0 0 0 0 0 0	48620250	1	17·3119	нет
8	230	1 0 0 0 0 0	194500845	4	41^2·5657	нет
9	232	1 0 1 1 1 2	2.85312E+11	5880	37^2·7124977	нет

Выбирать в качестве целевого не обязательно основное, а лишь достаточно низкое состояние: QAOA трудно сходится к глобальному минимуму, но при малой энергии подготовленного состояния вероятность измерить нужное собственное состояние высока. Эксперименты, описанные ниже, это подтверждают.

Аналогично приведены четыре низших собственных состояния для 3-кубитного случая (таблица Г.6). Sr-пары получаются из первых трёх уровней, включая основное. Второе возбуждённое состояние (000) эквивалентно оптимуму Бабаи; после квантовой оптимизации получены ещё более короткие векторы длиной 33 и 35, также дающие новые sr-пары. Целевым выбираем основное состояние (001), которое готовится в эксперименте. Для 10-кубитного случая (таблица Г.7) основное состояние (0100010010) ведёт к sr-паре и становится целевым.

Таблица Г.6 – Четыре низших уровня энергии и соответствующие квантовые состояния (3-кубитный случай).

уровни	**энергия**	**состояние**	u	v	$\lvert u - vN\rvert$	гладкое
0	33	0 0 1	1800	1	7×23	да
1	35	1 1 0	1944	1	17	да
2	36	0 0 0	2025	1	2^6	да
3	42	1 0 0	3645	2	277	да

Таблица Г.7 – Первые десять наименьших уровней энергии и соответствующие квантовые состояния для 10-кубитного случая. Основное состояние (0100010010) порождает гладкую пару отношений, а соответствующее значение $|u - vN| = 2 \cdot 31 \cdot 97 \cdot 109 \cdot 163 \cdot 433$ гладко по границе B_2, что делает данное состояние целевым для 10-кубитного случая.

уровень	**энергия**	**состояние**	u	v	$\lvert u - vN\rvert$	гладкое
0	51	0 1 0 0 0 1 0 0 1 0	785989264048241	3	$2\times31\times97\times109\times163\times433$	да
1	57	0 1 0 0 0 0 0 0 1 0	261933899831373	1	$2^3\times29\times203014633$	нет
2	60	0 0 0 0 0 0 0 0 0 0	262049748526566	1	$47\times139\times10523389$	нет
3	60	0 0 0 1 0 1 0 0 0 0	262123789565918	1	3803×3754673	нет
4	61	0 1 0 0 0 0 1 1 0 0	262027921960805	1	$2^4\times457\times2243\times2861$	нет
5	65	0 1 0 0 0 0 0 1 0 0	7598979238585630	29	$3^2\times211\times1531\times835897$	нет
6	66	0 0 0 0 0 1 1 0 0 0	4455399847833940	17	$2^3\times24407\times11764801$	нет
7	68	0 0 0 0 0 0 0 0 1 0	261988302332823	1	$2\times7^2\times22717\times22963$	нет
8	70	0 0 0 0 0 1 0 0 0 0	262012871275155	1	$2\times1693\times9412891$	нет
9	70	0 0 0 0 1 0 0 0 0 0	262002304109546	1	21304883317	нет

ПРИЛОЖЕНИЕ Д

Детали эксперимента

Параметры устройства

Эксперимент проводился на сверхпроводящем квантовом процессоре flip-chip-типа [44] с 10 кубитами (Q1–Q10) и 9 соединителями (C1–C9), попеременно расположенными в цепочечной топологии (рис. 2A). Все кубиты и соединители — трансмоны; их частоты настраиваются независимо при помощи медленных потоковых импульсов (длительностью до сотен микросекунд) либо быстрых Z-импульсов (до десятков микросекунд) по соответствующим управляющим линиям. Максимальные частоты кубитов (соединителей) $\approx$ 4.7 ГГц (9.0 ГГц), нелинейности $\approx$ -210 МГц (-150 МГц). Силе связи соседних кубитов регулируется от почти нуля до -10 МГц модуляцией частоты промежуточного соединителя. По управляющим линиям кубитов также подаются СВЧ-импульсы для однокубитных затворов. Времена релаксации, дефазировки, достоверности затворов и иные характеристики сведены в табл. Д.8.

Таблица Д.8 – Параметры устройства. ω_j^0 — это частота покоя кубита Q_j (idle frequency), при которой кубит инициализируется. η_j — это нелинейность кубита Q_j. $T_{1,j}$ и $T_{2,j}$ — соответственно время релаксации энергии и время дефазировки Рамсе кубита Q_j на частоте покоя. $F_{0,j}$ и $F_{1,j}$ — фиделити измерения кубита Q_j, подготовленного в состояниях $|0\rangle$ и $|1\rangle$ соответственно. e_j^S — одновременная ошибка Паули однокубитного затвора кубита Q_j. $e_{j,A(B)}^{\mathrm{CZ}}$ — одновременная ошибка Паули CZ-затвора кубитов Q_j и Q_{j+1} в группе A (B). ($\omega_j^{A(B)}$, $\omega_{j+1}^{A(B)}$) — оценочные частоты кубитов Q_j и Q_{j+1} в группе A (B) при выполнении CZ-затворов.

Кубит	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_{10}	среднее
$\omega_j^0/2\pi$ (GHz)	4.420	4.200	4.553	4.460	4.370	4.600	4.430	4.515	4.445	4.570	4.486
$\eta_j/2\pi$ (MHz)	-213	-209	-208	209	-211	-211	-209	-210	-211	-210	-210
$T_{1,j}$ (μs)	91.4	83.3	113.6	131.1	111.6	99.5	116.0	108.2	123.3	115.1	109.3
$T_{2,j}$ (μs)	5.3	7.0	5.7	4.3	6.2	4.9	5.8	8.2	6.9	5.9	6.2
$F_{0,j}$ (μs)	0.982	0.984	0.996	0.991	0.992	0.990	0.981	0.978	0.974	0.978	0.981
$F_{1,j}$ (μs)	0.949	0.967	0.942	0.957	0.951	0.958	0.960	0.958	0.927	0.923	0.949
ϵ_j^S (%)	0.11	0.07	0.09	0.09	0.15	0.12	0.07	0.08	0.09	0.09	0.09

Кубит	Q_1-Q_2	Q_3-Q_4	Q_5-Q_6	Q_7-Q_8	Q_9-Q_{10}	среднее
$(\omega_j^A, \omega_{j+1}^A)/2\pi$ (GHz)	4.315, 4.520	4.666, 4.460	4.600, 4.392	4.335, 4.540	4.570, 4.364	-
$\epsilon_{j,A}^{CZ}$ (%)	0.65	0.72	0.65	0.67	0.76	0.69

Кубит	Q_1	Q_2-Q_3	Q_4-Q_5	Q_6-Q_7	Q_8-Q_9	Q_{10}	среднее
$(\omega_j^B, \omega_{j+1}^B)/2\pi$ (GHz)	-	4.348, 4.553	4.510, 4.304	4.609, 4.400	4.532, 4.325	-	-
$\epsilon_{j,B}^{CZ}$ (%)	-	0.54	0.70	0.58	0.57	-	0.60

Калибровка экспериментальных затворов

Все кубиты инициализируются в основное состояние процедурой сброса: соединитель, соседний кубиту, резонансно сводится с ним на несколько наносекунд, что переносит возбуждение в соединитель; далее соединитель переводится на максимальную частоту, где его T_1 мало, и возбуждение быстро распадается. Шаги повторяются до полной инициализации. Затем реализуются экспериментальные схемы (рис. 2D). Для длинных пауз вставляются двойные $R_x(\pi)$-затворы, чтобы защитить кубит от дефазировки. После измерения сырых вероятностей строк битов проводится коррекция чтения [45].

Экспериментальная схема содержит затворы $R_x(\theta), R_y(\theta), R_z(\theta)$ (вращения на x, y, z на угол θ), Адамара и затворы с контролируемой фазой (CZ). R_x, R_y реализуются 30-нс СВЧ-импульсами с заданными фазами и амплитудами. R_z реализуются виртуальными Z затворами, которые могут считаться идеальными затворами, которые занимают нулевое время путем изменением фаз последующих импульсов [46]. Затвор Адамара реализуется как $H = R_y(\pi/2)R_z(\pi)$. CZ-затворы выполняются специально сформированными потоковыми импульсами, воздействующими на соседние кубиты и общий соединитель; детали описаны в [34; 47]. По возможности последовательные однокубитные затворы объединяются, уменьшая глубину схемы.

Здесь мы обсуждаем случай оптимизации фиделити затворов при выполнении CZ-затворов параллельно, что аналогично индивидуальной реализации, за исключением оптимизации параметров импульсов для соединителей. В соответствии с требованиями экспериментальной схемы, мы делим девять соединителей на две группы: $\{C_1, C_3, C_5, C_7, C_9\}$ для группы A и $\{C_2, C_4, C_6, C_8\}$ для группы B. Мы одновременно применяем прямоугольные магнитные импульсы с синусоидальной модуляцией вида $A(t) = z_{C_j} \cdot \left(1 - r_{C_j} + r_{C_j} \sin\left(\frac{\pi t}{t_{\text{gate}}}\right)\right)$ ко всем соединителям в группе A или B. Здесь z_{C_j} — это максимальная амплитуда потока, приложенного к соединителю C_j, r_{C_j} — модульный параметр, зафиксированный около 0,1, а $t_{\text{gate}} = 50$ нс. Обрати внимание, что 5-нс интервалы добавляются до и после этого магнитного импульса. Для оптимизации z_{C_j} мы подготавливаем кубиты Q_j и Q_{j+1} в состоянии $|11\rangle$ и применяем m циклов CZ-затворов при $m \in \{1, 3, 5\}$. Затем мы напрямую измеряем утечку в состояние $|2\rangle$ у кубита с более высокой резонансной частотой в паре, и минимизируем эту утечку для всех пар одновременно путём тонкой настройки всех z_{C_i}.

Мы используем кросс-энтропийный бенчмаркинг (XEB) [2] для оценки производительности наших квантовых затворов. Средняя ошибка Паули для одновременных однокубитных затворов составляет 0,09%. Средние ошибки Паули для CZ-затворов в группах A и B составляют 0,69% и 0,60% соответственно. Следует отметить, что структура экспериментальной схемы отличается от стандартных схем оценки производительности. В связи с этим мы используем альтернативные квантовые схемы для дополнительной верификации производительности наших CZ-затворов, которые имеют структуры, схожие со схемами, используемыми в нашем алгоритме (см. рис. Д.1). На рис. Д.1C показана точность XEB в зависимости от глубины схемы (количества циклов), демонстрируя ошибку цикла порядка 3,45%, на основе чего мы оцениваем среднюю ошибку CZ-затвора как 0,55%.

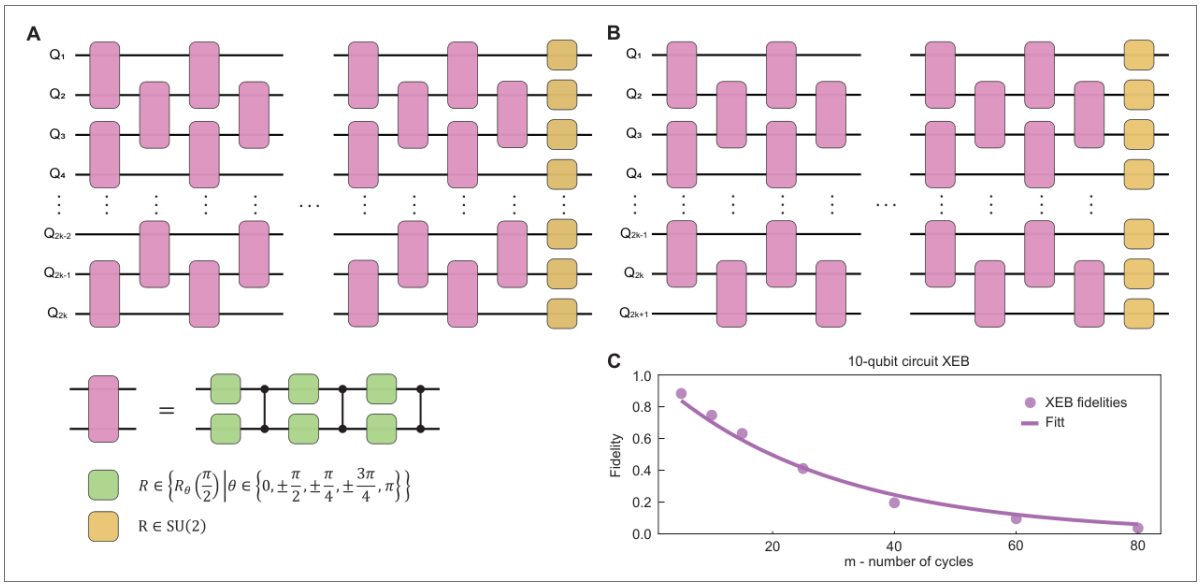

Рисунок Д.1 – Альтернативные квантовые схемы, используемые для бенчмаркинга CZ-затворов, для чётного (A) и нечётного (B) числа кубитов. Зелёные квадраты обозначают случайно выбранные полу-π-повороты вокруг восьми осей, где θ — угол между осью поворота и осью x. Жёлтые квадраты обозначают затворы, случайно выбранные из SU(2). C, фиделити XEB как функция числа циклов затвора для случая с 10 кубитами. Ошибка цикла аппроксимирована значением около 3,45 %, причём каждый цикл содержит слой из 11 однокубитных затворов, за которым в среднем следует слой из 4,5 CZ-затворов.

Процедура QAOA и сходимость

QAOA может находить приближённое основное состояние гамильтониана путём обновления параметров. Для QAOA с p слоями участвуют $2p$ вариационных параметров: $\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_p)$ и $\boldsymbol{\beta} = (\beta_1, ..., \beta_p)$. Основная задача квантового процессора — многократно подготавливать следующую параметрическую волновую функцию:

$$|\gamma, \beta\rangle = e^{-i\beta_p H_b} e^{-i\gamma_p H_c} \dots e^{-i\beta_1 H_b} e^{-i\gamma_1 H_c} |+\rangle^n, \tag{Д.53}$$

где $H_b = \sum_{j=1}^{n} \sigma_x^j$ — это смешивающий гамильтониан. Это состояние может быть подготовлено путём чередующегося применения унитарных операторов $U(H_c, \gamma) = e^{-i\gamma H_c}$ и $U(H_b, \beta) = e^{-i\beta H_b}$ с разными параметрами к состоянию равномерной суперпозиции $|+\rangle^n$. Классический оптимизатор используется для поиска оптимальных параметров $(\boldsymbol{\gamma}^*, \boldsymbol{\beta}^*)$, минимизирующих ожидаемое значение энергии проблемного гамильтониана:

$$E(\gamma, \beta) = \langle \gamma, \beta | H_c | \gamma, \beta \rangle \tag{Д.54}$$

Эта функция энергии может быть вычислена путём многократной подготовки волновой функции $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ в квантовом регистре и её измерения в вычислительном базисе. В результате получается квантовое состояние $|\boldsymbol{\gamma}^*, \boldsymbol{\beta}^*\rangle$, соответствующее приближённому решению. Алгоритм графически представлен на рис. 2C.

Здесь мы кратко представим классический оптимизатор, используемый в QAOA во время процедуры оптимизации параметров. Оптимизатор называется методом модельного градиентного спуска (model gradient descent, MGD) [48]. На самом деле, можно рассмотреть и множество других классических алгоритмов оптимизации, таких как метод симплекса Нелдера–Мида [49], квазиньютоновские методы [50; 51]. Производительность этих методов часто зависит от конкретной задачи. Показано как численно, так и экспериментально, что модельный градиентный спуск работает хорошо для некоторых вариационных квантовых анзацев [11; 48]. Ключевая идея MGD — использование модели для оценки градиента целевой функции, которая представляет собой непрерывную поверхность или гиперповерхность. Для оценки градиента в данной точке на поверхности случайным образом выбираются несколько близлежащих точек и

вычисляются значения целевой функции в этих точках. Затем квадратичная модель аппроксимирует поверхность этих точек с помощью метода наименьших квадратов. Градиент этой квадратичной модели используется в качестве замены истинного градиента, и алгоритм спускается по соответствующему направлению. Псевдокод приведён в алгоритме 3.

В нашем эксперименте метод MGD хорошо показал себя в трёх задачах факторизации. Он сходится к локальному или глобальному оптимуму в пределах 10 шагов из случайно выбранных начальных точек. При этом экспериментальные результаты сходимости сопоставимы с теоретическими на текущем масштабе. Подробности о траекториях сходимости приведены на рис. Д.2.

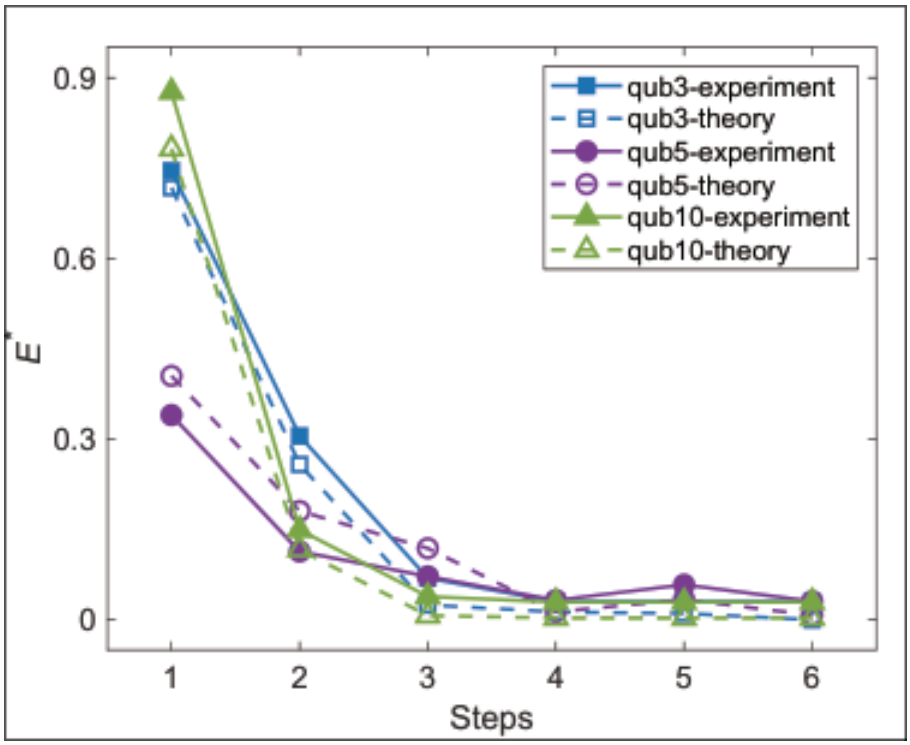

Рисунок Д.2 – Подробности о траекториях сходимости для трёх задач факторизации. Квадраты, кружки и треугольники соответствуют случаям с 3, 5 и 10 кубитами соответственно. Сплошные (пустые) символы обозначают экспериментальные (теоретические) результаты. По вертикальной оси отложено нормализованное значение энергетической функции E^*, а по горизонтальной оси — шаги вычислительной итерации.

Алгоритм 3 — Алгоритм Model Gradient Descent

Исходные параметры: начальная точка x_0, скорость обучения γ, радиус выборки δ, количество точек k, показатель убывания скорости α, постоянная устойчивости A, показатель убывания радиуса ξ, точность ε, максимальное число итераций n

Результат: оптимальное значение x

Инициализировать список L;
$x \leftarrow x_0$;
$m \leftarrow 0$;
до тех пор, пока *число вычислений функции* $+ k$ *не превышает* n **выполнять**
    Добавить кортеж $(x, f(x))$ в список L;
    $\delta' \leftarrow \delta/(m+1)^{\xi}$;
    Сэмплировать k точек равномерно случайно из δ'-окрестности x; пусть S — полученное множество;
    для каждого $x' \in S$ **выполнять**
        Добавить $(x', f(x'))$ в L;
    конец
    Инициализировать список L';
    для каждого *кортеж* $(x', y') \in L$ **выполнять**
        если $\|x' - x\| < \delta'$ **тогда**
            Добавить (x', y') в L';
        конец
    конец
    Аппроксимировать квадратичную модель по точкам из L' с помощью линейной регрессии с полиномиальными признаками;
    Пусть g — градиент квадратичной модели в x;
    $\gamma' \leftarrow \gamma/(m+1+A)^{\alpha}$;
    если $\gamma' \cdot \|g\| < \varepsilon$ **тогда**
        возвратить x;
    конец
    $x \leftarrow x - \gamma' \cdot g$;
    $m \leftarrow m + 1$;
конец
возвратить x;

ПРИЛОЖЕНИЕ Е

Постобработка: гладкие пары и линейные уравнения

Ниже представлены другие гладкие соотношённые пары, полученные в примерах факторизации чисел 1961, 48567227 и 261980999226229, как показано в следующих списках. В первом столбце указаны порядковые номера гладких соотношённых пар (sr-пар), а выражение $|u-vN|$ представлено в виде разложения по соответствующему простому базису. В случае с 3 кубитами мы приводим 20 независимых гладких соотношённых пар, и соответствующая булева матрица состоит из 20 векторов размерности 16. Следовательно, среди них обязательно найдётся группа линейно зависимых векторов, то есть система линейных уравнений имеет как минимум одно решение.

Случай с 3 кубитами

В таблице Е.9 приведён список булевых векторов, соответствующих показателям простого базиса, составленного из $u/(u-vN)$. Можно заметить, что четвёртый вектор является нулевым, что само по себе представляет собой вектор линейной зависимости. 10-й и 17-й векторы, а также 5-й и 16-й — это две группы линейно зависимых векторов. Другие линейно зависимые векторы необходимо определить, решая систему линейных уравнений. Каждая такая группа линейных зависимостей будет соответствовать квадратичному сравнению вида $X^2 \equiv Y^2 \pmod N$. С высокой вероятностью факторизация числа N может быть получена с помощью этого сравнения. Ниже мы приводим подробности факторизации $N = 1961$ с использованием конкретных гладких соотношённых пар.

Согласно приведённому выше обсуждению, из указанных гладких соотношённых пар можно найти решения системы линейных уравнений, например:

Пример 1: 4-я пара, где $u = 34 \cdot 5^2$, $v = 1$, $|u - vN| = 26$, что даёт квадратичное сравнение: $(9 \cdot 5)^2 - 8^2 = N$. Тогда: $p = \gcd(45+8, N) = 53, \quad q = \gcd(45-8, N) = 37$.

Пример 2: 9-я пара, где $u = 26 \cdot 5^2$, $v = 1$, $|u-vN| = 19^2$, $(8 \cdot 5)^2 + 19^2 = N$, и, согласно методу факторизации Гаусса, это приведёт к паре множителей:

$$p = x^2 + y^2, \quad q = a^2 + b^2. \tag{E.55}$$

Далее:

$$\begin{cases} |ax - by| = 40, \\ |bx + ay| = 19, \end{cases} \quad \text{или} \quad \begin{cases} |ax - by| = 19, \\ |bx + ay| = 40. \end{cases} \tag{E.56}$$

Решая уравнения, получаем: $a = 1$, $b = 6$, $x = 2$, $y = 7$. Подставляя в уравнение (E.55), получаем $p = 53$, $q = 37$. Или: $a = 2$, $b = 7$, $x = 6$, $y = -1$, тогда $p = 37$, $q = 53$.

Пример 3: Комбинация 10-й и 17-й пары даёт:

$$(2 \cdot 5^2 \cdot 2^2 \cdot 3^3)^2 \equiv (7 \cdot 11 \cdot 13)^2 \pmod{1961}. \tag{E.57}$$

Отсюда:

$$p = \gcd(5400 + 1001, 1961) = 37, \tag{E.58}$$

$$q = \gcd(5400 - 1001, 1961) = 53. \tag{E.59}$$

Пример 4: Комбинация 5-й и 16-й пары:

$$2^5 \cdot 5^2 \cdot 3 \cdot 5^4 \equiv 2 \cdot 4^3 \cdot 3^3 \cdot 4^3 \pmod{1961}, \tag{E.60}$$

то есть:

$$(2^2 \cdot 5^3)^2 \equiv (4^3 \cdot 3)^2 \pmod{1961}. \tag{E.61}$$

Следовательно:

$$p = \gcd(500 + 129, 1961) = 37, \tag{E.62}$$

$$q = \gcd(500 - 129, 1961) = 53. \tag{E.63}$$

Кроме того, простые множители также могут быть найдены с помощью решений линейных уравнений для других соотношений, которые здесь не приводятся.

Таблица E.9 – Булевы экспоненциальные векторы, соответствующие гладким соотношённым парам. Первый столбец представляет собой порядковый номер гладкой соотношённой пары (u, v). Второй столбец — это знак, который указывает на положительность или отрицательность выражения $u/(u - vN)$. Столбцы с третьего по семнадцатый представляют булевы показатели для первых 15 простых чисел базиса соответственно.

sn	**sign**	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}
1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0	0	0	0	0	1	0	1	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0
6	1	1	0	1	0	1	0	0	0	1	0	0	0	0	0	0
7	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0
9	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
10	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
11	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	1
12	1	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0
13	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
14	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	1	0	1	0	0	0	0	1	0	1	0	0	0	0	1	0
16	1	1	1	0	0	0	0	0	0	0	0	0	1	0	1	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	1
19	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0
20	0	0	1	0	1	0	0	1	0	0	0	0	1	0	1	0

Случай с 5 кубитами

В случае с 5 кубитами мы приводим 55 независимых гладких соотношённых пар в следующем списке. Соответствующая булева матрица содержит 55 векторов размерности 51 (50 измерений для простого базиса и 1 измерение для знака). Аналогично, среди них обязательно найдётся группа линейно зависимых векторов.

Из-за высокой размерности векторов, соответствующих случаю с 5 кубитами, мы приводим только одно решение системы линейных уравнений:

$$\begin{aligned} x = (&0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,0,1,0,0,\\ &1,1,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,\\ &0,0,0,0,0,0,1,1,1,0,0,0,0). \end{aligned} \tag{E.64}$$

Соответствующее решение для квадратичного сравнения:

$$\begin{aligned} X &= 639232456435359657331994419097900390625,\\ Y &= 121365727343256336293439260548453 04. \end{aligned} \tag{E.65}$$

Нетрудно проверить, что данное решение удовлетворяет уравнению:

$$X^2 \equiv Y^2 \pmod{N} \tag{E.66}$$

Кроме того, имеем:

$$\begin{aligned} p &= \gcd(X+Y, N)\\ &= \gcd(639232456435359657331994419097900390625+\\ &\quad 121365727343256336293439260548453 04,\ 48567227)\\ &= 7919,\\ q &= \gcd(X-Y, N)\\ &= \gcd(639232456435359657331994419097900390625-\\ &\quad 121365727343256336293439260548453 04,\ 48567227)\\ &= 6133. \end{aligned} \tag{E.67}$$

В результате мы получаем разложение числа на множители: $N = 48567227 = 7919 \times 6133$.

Случай с 10 кубитами

В случае с 10 кубитами мы приводим 221 независимую гладкую пару в Дополнительных материалах. Соответствующая булева матрица содержит 221 вектор размерности 201 (200 измерений для простого базиса и 1 измерение для знака). Мы приводим одно решение системы линейных уравнений:

$$
\begin{aligned}
x = (&1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,\\
&0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,\\
&1,1,0,0,0,1,0,0,0,0,1,0,0,0,0,1,1,0,0,0,0,1,0,1,\\
&1,0,1,1,0,0,1,0,0,0,1,1,1,0,0,0,1,0,1,1,1,0,1,0,\\
&1,0,0,1,0,0,0,0,1,1,1,1,0,1,1,0,1,0,1,0,0,1,0,0,\\
&1,1,0,0,0,1,1,0,0,0,0,1,0,0,1,1,1,0,1,0,0,0,0,1,\\
&0,0,1,0,1,1,1,1,0,1,0,0,0,0,1,0,1,0,0).
\end{aligned}
\tag{E.68}
$$

Соответствующее решение для квадратичного сравнения представлено в уравнении:

$$
\begin{aligned}
X = &75695763106501556705305764502754936587819598184067351577\\
&43303250785633285956794230102519018501335269995854539435\\
&01610097294132617819824833811012145914436530877754829641\\
&77181523394942294029136952806406322075276949827770146241 0\\
&83103264602165492969982376925586397260794635650709243752 0\\
&62898300577225894359044218428126425544707265515253992507 5\\
&22823290552782790244143095915540083328664735548909870120 3\\
&66526609754309643401619811910126985479577532456154006421 56\\
&60095214843750000000000000000000000000000000000000000,
\end{aligned}
\tag{E.69}
$$

$$
\begin{aligned}
Y = &89703025676439146996343189305085999464349858249769290536\\
&39263868715312151470596839682183291916841767330486102677\\
&25009214205336106668598949343065162224483352986497301 88\\
&13941971214083630347773876968401363896980941002118162 8\\
&14845846664464645570997588141798961920532615300189398 6\\
&17812364391639832172885099750660810556653762291758212 6\\
&73137514583348598029804401113412582240391388504667126 2\\
&19915873471668113404162973340975659170623801.
\end{aligned}
$$

Нетрудно проверить, что данное решение удовлетворяет следующему уравнению:

$$X^2 \equiv Y^2 \pmod{N} \tag{E.70}$$

Кроме того, имеем:

$$\begin{aligned} p &= \gcd(X + Y, N) = 15538213, \\ q &= \gcd(X - Y, N) = 16860433. \end{aligned} \tag{E.71}$$

В результате мы получаем разложение на множители:

$$N = 261980999226229 = 15538213 \times 16860433. \tag{E.72}$$

ПРИЛОЖЕНИЕ Ж

Исследование квантового преимущества

В этой части мы численно исследуем преимущество квантового оптимизатора по сравнению с алгоритмом Бабаи. Рассматриваемый критерий измерения — это качество коротких векторов для задачи ближайшего вектора (CVP). Качество короткого вектора положительно коррелирует с эффективностью получения гладких соотношённых пар в более эффективном методе факторизации Шнорра. Поскольку в настоящее время оценка аналитической сложности алгоритма QAOA остаётся открытым вопросом, в данном обсуждении предполагается, что процедура QAOA способна находить оптимальное решение задачи оптимизации за ограниченное время. Здесь мы используем параметр относительного расстояния r для измерения длины вектора вместо евклидовой нормы или квадратной нормы. Параметр определяется следующим образом:

$$r = \frac{\|\mathbf{b} - \mathbf{t}\|^2}{\det(\mathbf{B}'_{n,c})^{\frac{2}{n}}}, \tag{Ж.73}$$

где $B'_{n,c} = [B_{n,c},\ Nc]$. Этот параметр использует $2/n$-степень определителя расширенной решётки $B'_{n,c}$ для измерения относительной длины короткого вектора $b - t$, что позволяет в определённой степени уменьшить влияние различий в определителе решёток на качество коротких векторов.

Результаты на случайных выборках

Сначала мы исследуем производительность квантового оптимизатора и классического алгоритма Бабаи на случайных выборках задачи CVP. Здесь мы генерируем 50 случайных примеров CVP (решётка и целевой вектор) при условиях размерности решётки $n = 7$, точности $c = 10$ и $n = 10$, $c = 7$ соответственно. Для каждой случайной выборки определитель решётки и целевой вектор остаются одинаковыми — меняются только элементы главной диагонали решётки, которые случайным образом переставляются. Результаты представлены на рис. Ж.3. По горизонтальной оси отложены случайные выборки, по вертикальной — относительное качество r результирующего вектора. Синие

(жёлтые) столбцы представляют результаты классического алгоритма Бабаи (квантовой оптимизации). Как видно на графике, результаты, полученные с помощью квантовой оптимизации, не хуже классических, а во многих случаях существенно лучше, то есть получены более короткие векторы.

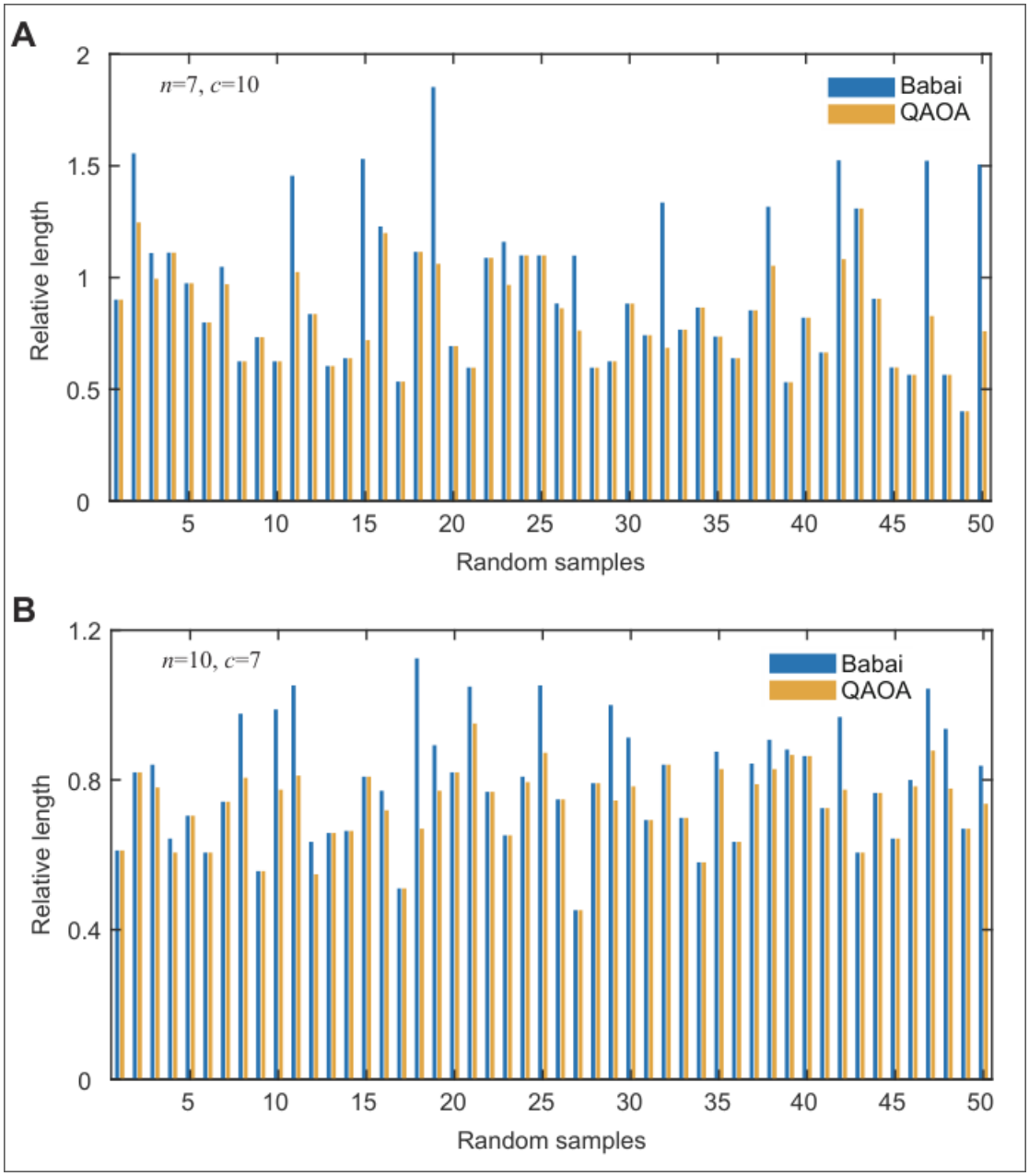

Рисунок Ж.3 – Производительность на случайных выборках для квантового оптимизатора (QAOA) по сравнению с алгоритмом Бабаи. A (B) — результаты для 50 случайных примеров задачи CVP при условиях $n = 7$, $c = 10$ ($n = 10$, $c = 7$). Жёлтые (синие) столбцы представляют результаты квантового оптимизатора (классического алгоритма Бабаи). Можно наблюдать, что в ряде случаев результаты квантовой оптимизации превосходят классические.

Квантовое преимущество и точность решётки

Мы дополнительно исследуем преимущество квантового оптимизатора при увеличении параметра точности c решётки. На рис. Ж.4А представлены численные результаты при увеличении параметра c от 5 до 14 для размерностей решётки $n = 12$ и $n = 14$ соответственно. Результаты усреднены по 40 случайно сгенерированным примерам задачи CVP для каждого набора параметров $\{n, c\}$. Символы в виде кружков (треугольников) соответствуют результатам при $n = 12$ ($n = 14$). Сплошные (пустые) символы обозначают результаты квантового (классического) алгоритма. Погрешности отображают доверительный интервал при стандартном отклонении, равном единице.

В обоих случаях, при $n = 12$ и $n = 14$, после квантовой оптимизации получаются более короткие векторы. Рассматривая случай $n = 14$ в качестве примера, можно увидеть, что разрыв в качестве между результатами алгоритма Бабаи и квантового алгоритма постепенно увеличивается с ростом параметра c. Это указывает на то, что качество вектора после квантовой оптимизации в среднем выше, чем у алгоритма Бабаи, при увеличении определителя решётки.

Кроме того, на рис. Ж.4В приведена доля выборок, где квантовая оптимизация дала преимущество, относительно всех 40 случайных примеров. Эти доли показаны синими и оранжевыми столбцами для случаев $n = 12$ и $n = 14$ соответственно. Мы обнаружили, что для $n = 12$ доля квантового преимущества составляет примерно 0,5, а для $n = 14$ она увеличивается до 0,65. Результаты свидетельствуют о том, что квантовое преимущество становится более заметным при увеличении размерности решётки. Эти результаты будут дополнительно подтверждены в следующем разделе.

Квантовое преимущество и размерность решётки

Здесь мы исследуем зависимость преимущества квантового оптимизатора от размерности решётки до $n = 14$. Для каждой размерности n результаты усреднены по 40 случайным выборкам при $c = n$. Как показано на рис. Ж.5А, качество вектора после квантовой оптимизации в среднем выше, чем при использовании классического алгоритма Бабаи.

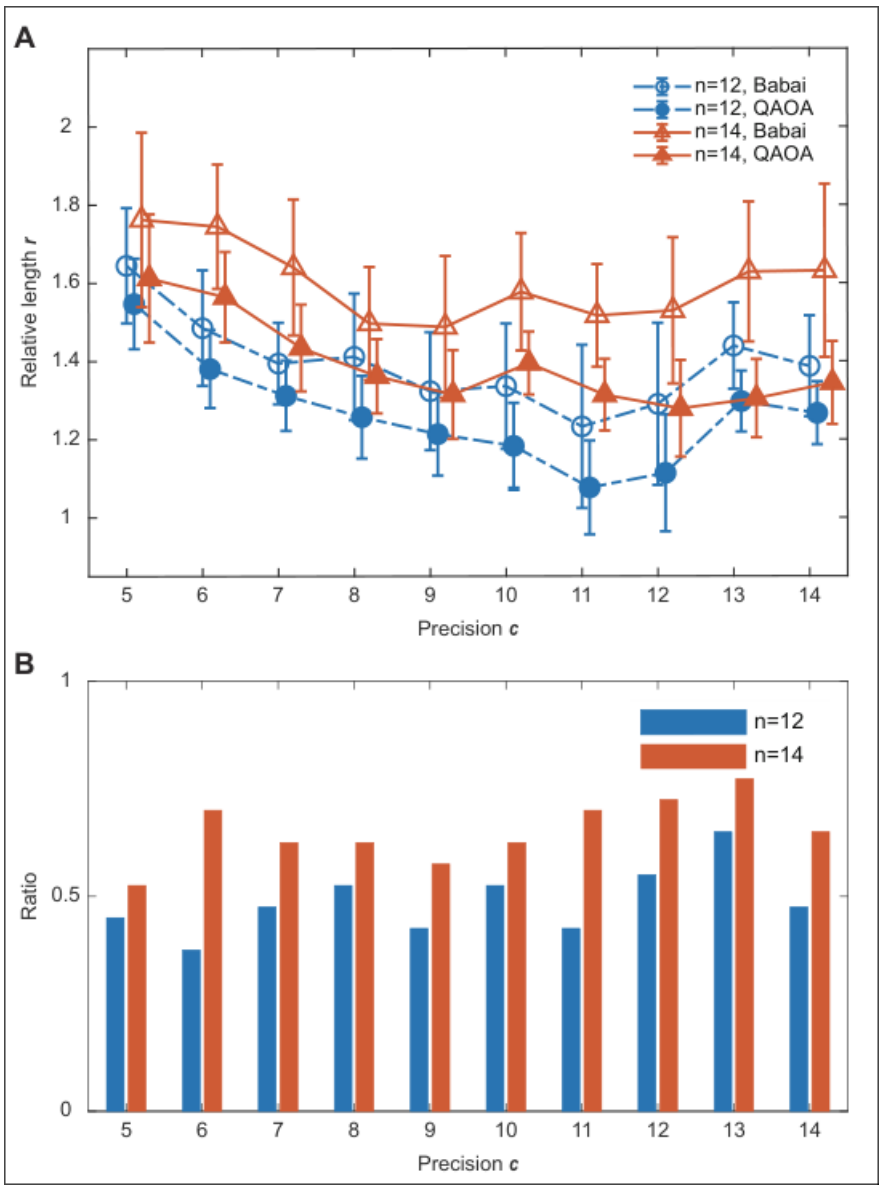

Рисунок Ж.4 – Производительность квантового оптимизатора при увеличении точности решётки. A: Относительные длины для квантовых и классических методов. По горизонтальной оси отложен параметр точности c, положительно коррелирующий с определителем решётки. B: Доля квантового преимущества для 40 случайных выборок, синие (оранжевые) столбцы для случая $n = 12$ ($n = 14$). На графике видно, что в среднем квантовая оптимизация даёт лучшие результаты, особенно при больших значениях определителя (связанного с c).

Иначе говоря, с помощью квантовой оптимизации можно найти более короткий вектор. Разрыв в качестве между квантовыми и классическими результатами становится более заметным с увеличением размерности решётки, что означает, что преимущество квантового метода возрастает в более крупных системах. Также мы подсчитали долю квантового преимущества среди 40 случайных выборок, представленных на рис. Ж.5B. Эта доля возрастает с уве-

личением размерности, что согласуется с различием тенденций на графиках качества векторов на рис. Ж.5А. Оба результата указывают на то, что преимущества квантовых методов будут становиться всё более очевидными с ростом размерности решётки.

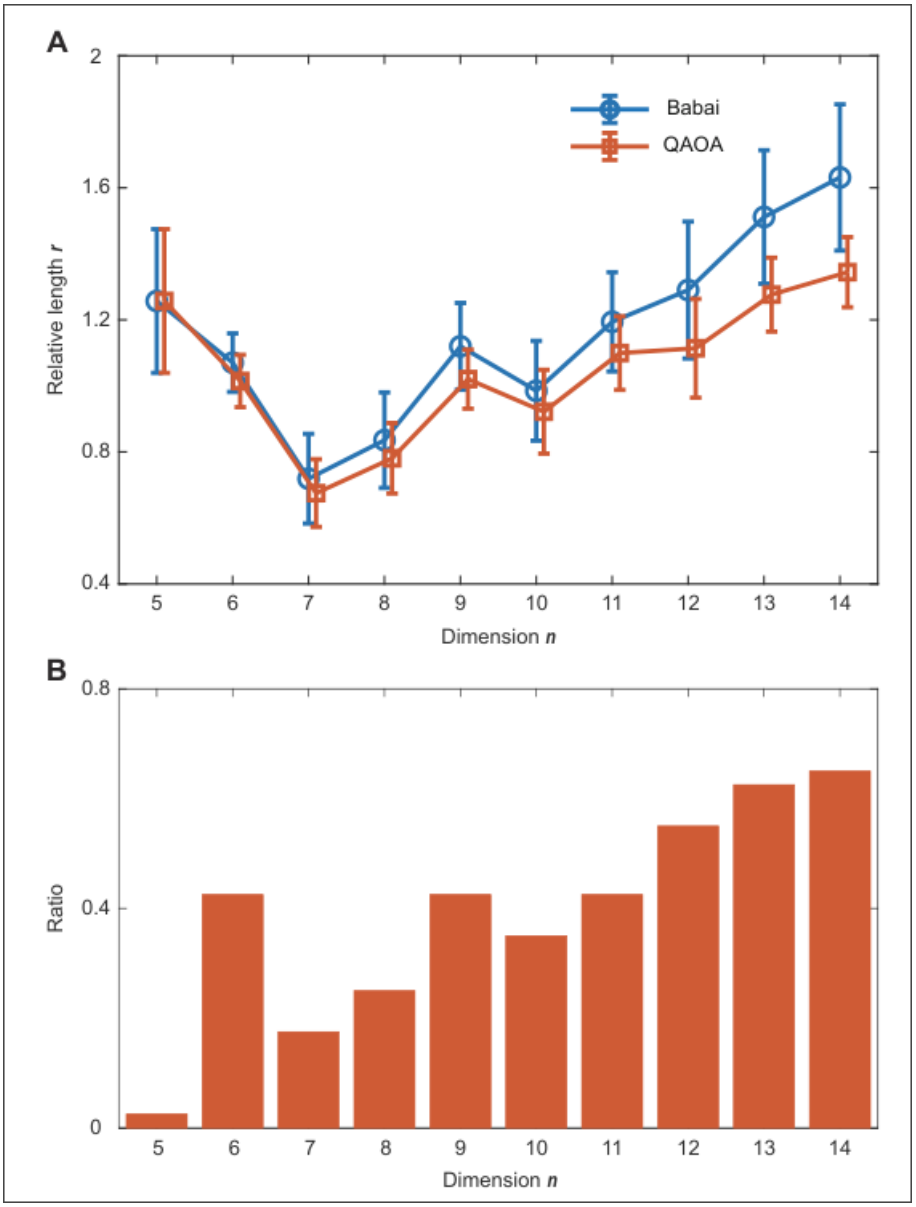

Рисунок Ж.5 – Производительность квантового оптимизатора при увеличении размерности решётки. А: Относительная длина векторов для квантового и классического методов. По горизонтальной оси отложена размерность решётки n. Результаты усреднены по 40 случайным выборкам при $c = n$. Погрешности отображают доверительный интервал при стандартном отклонении, равном единице. В: Доля квантового преимущества среди 40 случайных выборок. Можно заметить, что разрыв в относительном расстоянии между квантовыми и классическими результатами увеличивается с ростом размерности решётки.

ПРИЛОЖЕНИЕ З

Оценка ресурсов для RSA-2048

Введение

Сколько квантовых ресурсов требуется для факторизации 2048-битного RSA-числа? В этом разделе мы сосредотачиваемся на конкретных квантовых ресурсах, необходимых для факторизации 2048-битного RSA-числа на основе алгоритма SQIF. Рассматриваемые квантовые ресурсы в основном включают количество физических кубитов и глубину схемы QAOA с одним слоем. Обычно квантовые схемы не могут быть непосредственно выполнены на квантовых вычислительных устройствах, поскольку при их разработке не учитываются особенности связности кубитов или топологии реальных физических систем. Процесс выполнения часто требует дополнительных квантовых ресурсов, таких как вспомогательные (ancilla) кубиты и увеличение глубины схемы. Мы обсуждаем необходимые квантовые ресурсы для факторизации реальных RSA-чисел в терминах трёх типов топологий: полной графовой топологии (K_n), двумерной решётки (2DSL) и одномерной цепочки (LNN). Мы демонстрируем с использованием конкретных схем, что процесс внедрения (embedding) не требует дополнительных кубитов. Более того, глубина схемы QAOA с одним слоем линейно зависит от размерности n квантовой системы для всех трёх топологий. Таким образом, для факторизации целых чисел с использованием алгоритма SQIF мы потребляем сублинейное количество квантовых ресурсов. В качестве примера рассмотрим RSA-2048. Необходимое количество кубитов оценивается как $n = \frac{2 \cdot 2048}{\log 2048} \approx 372$. Глубина квантовой схемы QAOA с одним слоем составляет 1118 для топологии K_n, 1139 для топологии 2DSL и 1490 для самой простой топологии LNN. Эти значения достижимы для NISQ-устройств в ближайшем будущем или даже уже сегодня.

Описание проблемы

Сначала рассмотрим построение гамильтониана H_c. Используя правила однокубитного кодирования, соответствующий гамильтониан H_c может быть представлен как двумерная изинговская модель следующего вида:

$$H_c = \sum_{i=1}^{n} h_i \sigma_z^i + \sum_{i,j=1}^{n} J_{i,j} \sigma_z^i \sigma_z^j, \tag{3.74}$$

где параметры h_i, $J_{i,j}$ определяются коэффициентами линейных и квадратичных членов задачи квадратичной безусловной бинарной оптимизации (QUBO). Символ суммы в правой части второго выражения пробегает все комбинации индексов. Если рассматривать каждый квадратичный член как ребро в неориентированном графе, то все ZZ-члены $\{\sigma_i^z \sigma_j^z\}_{i<j}$ формируют полный граф порядка n, то есть граф K_n. Иначе говоря, топология связности логических кубитов — это граф K_n. В качестве примеров можно рассмотреть случаи с 3 кубитами и 5 кубитами, приведённые в основном тексте: топология кубитов проблемного гамильтониана представляет собой соответственно полный граф порядка 3 и 5, как показано на рис. З.6A и B.

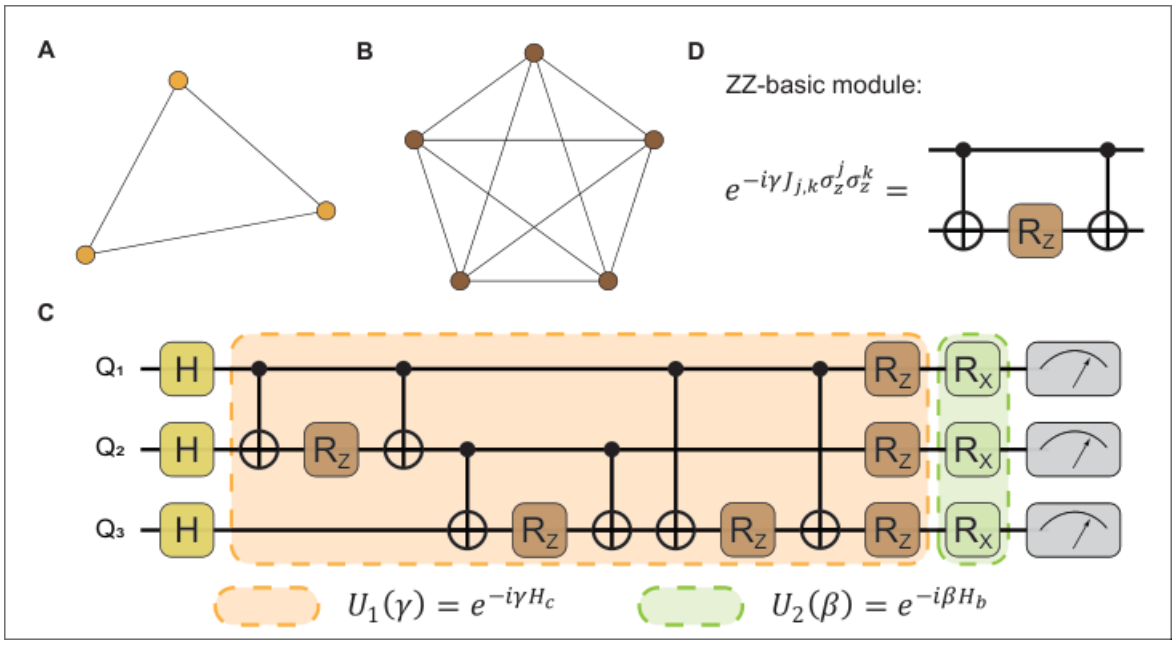

Рисунок З.6 – Связность кубитов и типичная квантовая схема QAOA. A — топология связности кубитов для случая с 3 кубитами, представляющая собой граф K_3. B — случай с 5 кубитами, соответствующий графу K_5. C — типичная квантовая схема QAOA с одним слоем для гамильтониана QUBO на 3 кубитах. Она в основном состоит из двух операторов: $U_1(\gamma)$ и $U_2(\beta)$, соответствующих операторам эволюции проблемного гамильтониана H_c и смешивающего гамильтониана H_b соответственно. D — базовый модуль ZZ. Он состоит из двух CNOT-затворов и одного однокубитного вращения R_z, расположенных по схеме «бутерброда».

Типичная схема QAOA для гамильтониана типа K_n показана на рис. З.6C. В квантовой схеме однослойной итерации QAOA участвуют два типа унитарных операторов. $U_1(\gamma)$ — оператор эволюции проблемного гамильтониана H_c, а $U_2(\beta)$ — оператор эволюции смешивающего гамильтониана H_b, состоящего из однокубитных вращений вокруг оси x. Оператор $U_1(\gamma)$ можно реализовать с помощью однокубитных вращений R_z и двухкубитных блоков, соответствующих локальным (линейным) членам и ZZ-членам гамильтониана H_c. Основное внимание мы уделяем двухкубитным блокам, которые определяются следующим образом:

$$\mathrm{ZZ}_{j,k}(\gamma) = e^{-i\gamma J_{j,k}\sigma_z^j\sigma_z^k}. \tag{З.75}$$

Унитар $\mathrm{ZZ}_{j,k}(\gamma)$ можно реализовать с помощью комбинации двух CNOT-затворов и одного R_z-затвора, образующих «бутерброд», как показано на рис. З.6D. В дальнейших обсуждениях мы будем рассматривать такую комбинацию затворов как базовый модуль глубины 3 без учёта его компиляции в конкретной физической системе. Физическая система, как правило, компилирует этот модуль на основе своей нативной универсальной системы квантовых затворов, и это обычно приводит к добавлению не более чем $O(1)$ дополнительных операций и увеличению глубины схемы.

Так как глубина схемы для однокубитных операций в $U_1(\gamma)$ и $U_2(\beta)$ не превышает 2 в физической реализации, мы не будем их далее обсуждать. Мы сосредотачиваемся на задаче внедрения квадратичных членов из $U_1(\gamma)$ в физическую систему. В частности, мы оцениваем накладные затраты по числу кубитов и глубине схемы после внедрения группы ZZ-членов, образующих схему типа K_n. В дальнейшем эта задача будет называться задачей внедрения типа K_n (Kn-type embedding problem).

Глубина схемы при топологии полного графа

Сначала рассмотрим идеальный случай, при котором любые два кубита могут напрямую взаимодействовать. Иначе говоря, топология связности кубитов в физической системе представляет собой полный граф. В этом сценарии задача внедрения схемы типа K_n не требует дополнительных кубитов или квантовых SWAP-затворов. Следовательно, глубина квантовой схемы может быть минимизирована.

Полный граф K_n содержит $n(n-1)/2$ рёбер, что означает, что схема QAOA содержит $O(n^2)$ базовых модулей ZZ. Глубина однослойной схемы QAOA без оптимизации составляет $O(n^2)$. Поскольку ZZ-члены в операторе $U_1(\gamma)$ коммутируют между собой, мы можем произвольно переставлять порядок двухкубитных взаимодействий, чтобы минимизировать глубину схемы.

Здесь мы представляем схему оптимизации, основанную на теории максимального паросочетания в неориентированном графе, которая позволяет уменьшить глубину схемы до $O(n)$. Эта схема является оптимальной, если рассматривать ZZ-член как базовый модуль.

Определение 2. Паросочетание и максимальное паросочетание. Обозначим неориентированный граф как $G(V, E)$ и пусть $M \subseteq E(G)$ — подмножество рёбер, такое что для всех пар $(e_i, e_j) \in M$ рёбра e_i и e_j не смежны в G. То есть рёбра в M не имеют общих вершин. Тогда M называется паросочетанием в графе G. Для каждого ребра $e = (u, v)$ в паросочетании M, говорят, что ребро e и вершины u, v покрыты паросочетанием M. Каждая вершина в графе либо не покрыта M, либо покрыта ровно одним ребром из M. Если не существует другого паросочетания M' в G, такого что $|M'| > |M|$, то M называется максимальным паросочетанием в G. Если каждая вершина в G покрыта паросочетанием M, то M называется совершенным паросочетанием в G.

Согласно определению, совершенное паросочетание обязательно является максимальным. Количество максимальных паросочетаний в полном графе определяется следующей леммой.

Лемма 3 (Максимальное паросочетание в полном графе) Существует $2n-1$ совершенных паросочетаний без повторяющихся рёбер в полном графе чётного порядка K_{2n}. Аналогично, существует $2n-1$ максимальных паросочетаний без повторяющихся рёбер в полном графе нечётного порядка K_{2n-1}.

Доказательство. Для полного графа чётного порядка K_{2n} расположим вершины $v_1, v_2, \ldots, v_{2n-1}$ по окружности в виде $(2n-1)$-угольника и добавим вершину v_{2n} в центр. Выберем произвольную вершину v_i, где $1 \leq i \leq 2n-1$, и построим паросочетание M_i, включающее ребро (v_i, v_{2n}) и все рёбра, перпендикулярные ему. Легко доказать, что M_i — совершенное паросочетание для любого i, и ни одно ребро не повторяется между M_i и M_j, если $i \neq j$. Таким образом, в полном графе K_{2n} существует $2n-1$ совершенных паросочетаний без повторений. Так как в графе K_{2n} всего $n(2n-1)$ различных рёбер, и $2n-1$ совершенных паросочетаний уже покрывают их без повторений, других возможных неповто-

ряющихся паросочетаний не существует. Для случая нечётного полного графа K_{2n-1} добавим вспомогательную вершину v_{2n}, тем самым превратив задачу в случай чётного порядка. Затем в каждом совершенном паросочетании удалим ребро (v_i, v_{2n}). В результате получим $2n-1$ максимальных паросочетаний для нечётного случая. Это завершает доказательство.

Утверждение 3

Если связность кубитов в физической системе представляет собой полный граф, то гамильтониан типа K_n может быть внедрён без дополнительных кубитов, а глубина внедрённой квантовой схемы составляет $O(n)$.

Доказательство. Согласно определению паросочетания (см. Определение 2), рёбра в одном паросочетании не имеют общих вершин. Следовательно, соответствующие двухкубитные взаимодействия можно выполнять одновременно и параллельно. Предположим, что каждый ZZ-член компилируется с использованием базового модуля ZZ (см. рис. S7D), тогда глубина схемы для одного паросочетания равна 3. Согласно Лемме 3, если n чётно, то в полном графе K_n существует $n-1$ неповторяющихся совершенных паросочетаний, покрывающих все рёбра. Значит, можно построить квантовую схему из $n-1$ слоёв, где каждый слой содержит $n/2$ модулей ZZ, выполняемых параллельно. Общая глубина схемы составит $3(n-1)$. Если n нечётно, то по Лемме 3 в полном графе K_n существует n максимальных паросочетаний без повторяющихся рёбер. Каждое такое паросочетание содержит $(n-1)/2$ рёбер, и все вместе они покрывают все рёбра графа K_n. Значит, можно построить квантовую схему из n слоёв, каждый из которых выполняет $(n-1)/2$ двухкубитных операций параллельно. Общая глубина схемы составляет $3n$. Таким образом, схема типа K_n может быть реализована без дополнительных кубитов, а глубина внедрённой квантовой схемы составляет $O(n)$. Доказательство завершено.

Доказательство Леммы 3 предоставляет точную конструкцию каждого совершенного паросочетания в полном графе K_n, а значит — и точную схему построения внедрённой квантовой схемы с глубиной $O(n)$. Схема оптимальна, если рассматривать ZZ-члены в качестве базового модуля. В таком случае количество ZZ-операций в каждом слое максимально, то есть достигается наивысшая степень параллелизма. Поскольку все максимальные (совершенные) паросочетания покрывают все рёбра без повторений, схема является оптимальной.

Связность кубитов — это ценный ресурс в устройствах NISQ. Как правило, построение полносвязной топологии в масштабируемых квантовых системах затруднено. Тем не менее, она достижима в некоторых особых системах, таких как ионные ловушки [52], оптические квантовые системы и системы с большой квантовой памятью [15].

Глубина схемы при линейной и решётчатой топологиях

Линейная цепочка (linear chain) — одна из самых распространённых топологий, которая относительно легко реализуется в реальных квантовых системах. Эта топология, также известная как архитектура линейного ближайшего соседа (LNN), предполагает расположение кубитов на линии с возможностью взаимодействия только между соседними кубитами. В этом разделе мы рассматриваем ресурсы по числу квантовых затворов и глубине схемы при внедрении гамильтониана типа K_n в одномерную систему LNN. Результаты для двумерной решётки (lattice system) можно сформулировать как следствие.

Задача внедрения произвольных топологий гамильтонианов в LNN широко исследуется [53—59], и уже существуют зрелые методы. В 2007 году Донни Чеунг и др. [55] исследовали накладные расходы при преобразованиях между различными топологиями на основе графовых моделей. Они указали, что отображение произвольной схемы в LNN требует не более $O(n)$ дополнительной глубины на основе параллельной сортировки. Однако конкретная схема преобразования гамильтониана типа K_n в LNN ими не приводится. В 2009 году Юити Хирата предложил эффективный метод отображения произвольных квантовых схем в LNN на основе идеи пузырьковой сортировки [56]. В 2021 году исследователи из Google применили параллельную пузырьковую сортировку для выполнения внедрения из K_{17} в LNN и провели соответствующие эксперименты на сверхпроводниковом квантовом процессоре Sycamore [11]. В целом, отображение K_n в LNN требует $O(n^2)$ операций SWAP и $O(n)$ дополнительной глубины схемы при использовании параллельной пузырьковой сортировки. Ниже мы приведём независимое доказательство этого результата на основе параллельного пузырькового алгоритма.

Чтобы внедрить полный граф K_n в LNN, необходимо выполнять дополнительные SWAP-операции для перестановки (или сортировки) кубитов в нужном порядке. Минимизация числа SWAP-затворов является ключевой задачей. Согласно работе [56], сеть SWAP-операций пузырьковой сортировки оптимальна для этой цели. Следовательно, справедлива следующая лемма.

Лемма 4. Пусть $x_1, x_2, \ldots, x_n$ — начальный порядок n кубитов в архитектуре LNN. Рассмотрим перестановку, изменяющую порядок на $x_{j_1}, x_{j_2}, \ldots, x_{j_n}$. Минимальное число необходимых SWAP-затворов эквивалентно числу операций обмена в алгоритме пузырьковой сортировки.

Лемма 4 показывает, что пузырьковая сортировка является оптимальной для перестановки порядка кубитов в случае, когда возможно взаимодействие только между соседними. Таким образом, квантовая схема SWAP-перестановок эквивалентна классической пузырьковой сортировке.

Пузырьковая сортировка начинается с головы массива данных, сравнивает первые два элемента и, если первый больше второго, меняет их местами. Процесс повторяется для каждой пары соседних элементов, пока не будет достигнут конец массива. Повторяется до тех пор, пока за один проход не произойдёт ни одной перестановки. Каждая пара сравнивается только один раз, поэтому среднее и худшее время выполнения составляет $O(n^2)$.

Рассмотрим худший случай: начальный порядок $1, 2, \ldots, n$, и требуется получить обратный порядок $n, n-1, \ldots, 1$. Согласно лемме 4, пузырьковая сортировка требует наименьшего числа SWAP-затворов. В этом случае необходимо $n(n-1)/2$ обменов, что в точности покрывает все рёбра полного графа K_n. Следовательно, классическая сеть пузырьковой сортировки, реализующая обратную перестановку, эквивалентна внедрению гамильтониана типа K_n в LNN.

Параллельная пузырьковая сортировка позволяет сократить время выполнения до $O(n)$. Основная идея — одновременное сравнение всех соседних пар входных данных с чередованием между нечётной и чётной фазами. Псевдокод приведён в Алгоритме 4. При размере входных данных n алгоритм выполняет n итераций. Каждая итерация делится на чётную и нечётную фазы в зависимости от номера шага. Если n нечётно, каждая фаза выполняет $(n-1)/2$ операций сравнения и обмена, которые можно проводить параллельно. Если n чётно, то нечётная и чётная фазы выполняют $n/2$ и $n/2-1$ операций соответственно.

Следующая лемма справедлива для алгоритма параллельной пузырьковой сортировки.

Алгоритм 4 — Параллельная пузырьковая сортировка

Исходные параметры: Data, массив из n элементов
Результат: Data в обратном порядке

цикл $i \leftarrow 1$ **от** n **выполнять**
  Flag $\leftarrow \mathrm{mod}(i, 2)$; ; `// флаг для чётной и нечётной фазы`
  цикл $j \leftarrow 1$ **от** $\left\lfloor \frac{n-1+Flag}{2} \right\rfloor$ **выполнять**
    если $Data[2j - Flag] < Data[2j + 1 - Flag]$ **тогда**
      swap Data[$2j$ − Flag] и Data[$2j + 1$ − Flag];
    конец
  конец
конец

Лемма 5. Пусть n — размер входных данных, а $p \leq \lfloor n/2 \rfloor$ — число доступных процессоров. Тогда временная сложность алгоритма составляет $O(n^2/2p)$. Минимум достигается при $p = \lfloor n/2 \rfloor$ с не более чем n итерациями.

Если количество процессоров меньше $n/2$, то один процессор выполнит примерно $\lfloor n/2p \rfloor$ операций сравнения и обмена во внутреннем цикле каждой фазы. В этом случае сложность алгоритма составляет $O(n^2/2p)$. При этом оптимальное количество процессоров — $\lfloor n/2 \rfloor$, и каждый процессор во внутреннем цикле выполняет не более одной операции. Внешний цикл алгоритма завершит всю пузырьковую сортировку за не более чем n итераций. В квантовых вычислениях предполагается, что устройство обладает достаточным числом управляющих процессоров, чтобы выполнять двухкубитные операции параллельно, если они не используют один и тот же кубит. Отсюда следует следующий результат.

Утверждение 4. Схема типа K_n может быть внедрена в физическую систему с топологией LNN без дополнительных кубитов, и глубина внедрённой квантовой схемы составляет $O(n)$.

Доказательство. Пусть начальный порядок кубитов в системе LNN: $1, 2, \ldots, n$, и он перестраивается в обратный порядок: $n, n-1, \ldots, 1$. Пузырьковая сортировка требует $n(n-1)/2$ перестановок. Сеть обменов пузырьковой сортировки точно покрывает все рёбра полного графа K_n, тем самым реализуя внедрение схемы K_n в LNN. Этот процесс реализуется параллельной пузырьковой схемой Γ с $n/2$ процессорами. Согласно лемме 5, схема требует не более n итераций, и на каждой итерации выполняется $\lfloor n/2 \rfloor$ обменов парал-

лельно. Конкретная параллельная сеть SWAP-операций, реализующая данный процесс, показана на рис. З.7. Следовательно, внедрённую схему в системе LNN можно построить, заменив SWAP-затвор в Г на базовый модуль ZZ-SWAP (см. рис. S8C). Так как глубина модуля ZZ-SWAP составляет 4, то общая глубина схемы — $4n$. Весь процесс внедрения не требует дополнительных кубитов. Доказательство завершено.

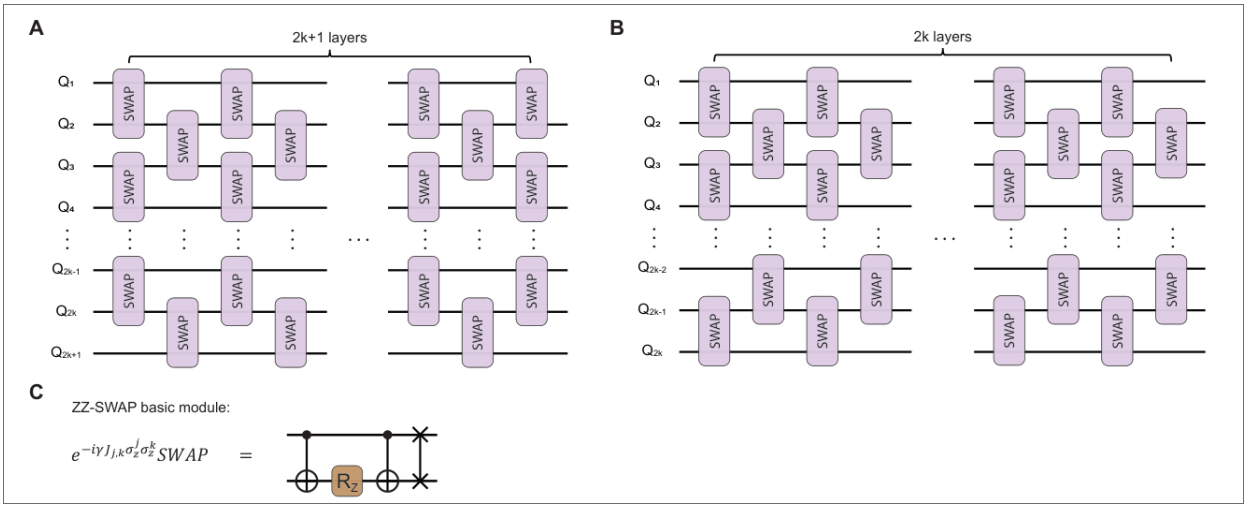

Рисунок З.7 – Схема параллельной пузырьковой сортировки, реализующей обратную перестановку. А — случай нечётного числа кубитов, В — случай чётного числа кубитов. В обоих случаях количество слоёв внешнего цикла составляет n, что приводит к глубине схемы $O(n)$. С — базовый модуль ZZ-SWAP: это квантовая схема глубины 4, представляющая собой операцию SWAP, выполненную после базового модуля ZZ.

Кроме того, поскольку каждый ZZ-член в K_n становится модулем ZZ-SWAP, требуется дополнительная нагрузка в виде $n(n-1)/2$ операций SWAP и глубины n после внедрения. После выполнения схемы порядок кубитов окажется обратным, и его можно восстановить, снова запустив схему QAOA.

Система с решётчатой топологией (lattice topology) также известна как двумерная квадратная решётка (2DSL), в которой кубиты располагаются в виде двумерной решётки, и разрешены только соседние взаимодействия. Поскольку одномерная цепочка может быть непосредственно вложена в двумерную решётку (например, вдоль одной из диагоналей или рядов), глубина внедрённой схемы для гамильтониана типа K_n не будет превышать глубину в случае LNN. Следовательно, справедливо следующее следствие.

Следствие 1. Глубина схемы для гамильтониана типа K_n, внедрённой в двумерную квадратную решётку (2DSL), составляет $O(n)$ и не требует дополнительных кубитов.

Топология полного графа является идеальной топологией: схема гамильтониана типа K_n может быть внедрена оптимально по глубине $O(n)$ без каких-либо дополнительных квантовых ресурсов. Для топологии LNN необходима дополнительная глубина $O(n)$. Поскольку схема внедрения на основе параллельной пузырьковой сортировки также имеет сложность $O(n)$, она является оптимальной как для LNN, так и для 2DSL-систем в смысле нотации $\mathcal{O}$. Кроме того, в работе [55] предложен более эффективный метод внедрения гамильтониана типа K_n в систему 2DSL с дополнительной глубиной $O(\sqrt{n})$, что достигается за счёт удобной двумерной структуры.

Оценка ресурсов для факторизации RSA-2048

В этом разделе мы обсуждаем необходимые квантовые ресурсы для атаки на реальные RSA-числа на основе полученных выше результатов. Поскольку внедрение схемы не требует дополнительных кубитов, количество кубитов, необходимое для факторизации m-битного числа, оценивается как $n = \frac{2m}{\log m}$ (здесь параметр точности $c = 1$). Глубина внедрённой схемы для гамильтониана типа K_n составляет $3n$ для полностью связанной системы, $4n$ — для системы с линейной топологией (LNN) и $3n + \sqrt{n}$ — для двумерной решётки (2DSL), согласно Donny Cheung и соавт. [55]. Эти оценки получены без учёта особенностей компиляции базового модуля ZZ (или ZZ-SWAP) в конкретной физической реализации. В качестве примера рассмотрим RSA-2048. Число кубитов: $\frac{2 \cdot 2048}{\log 2048} \approx 372$. Глубина схемы однослойного алгоритма QAOA составляет приблизительно $3n + 2 = 1118$ в полностью связанной системе, где 2 дополнительных уровня приходятся на однокубитные операции: один уровень для R_z-вращений и один — для R_x-вращений. а для двумерной квадратной решётки (2DSL) — около $3n + \sqrt{n} + 2 = 1139$. Необходимые квантовые ресурсы для факторизации RSA-чисел различной длины приведены в Таблице З.10.

Таблица З.10 – Оценка квантовых ресурсов для RSA-чисел. Основные квантовые ресурсы, рассматриваемые здесь, — это количество кубитов и глубина квантовой схемы однослойного алгоритма QAOA в трёх типичных топологиях: полностью связанной системе (K_n), двумерной решётке (2DSL) и одномерной цепочке (LNN). Полученные результаты не учитывают особенности компиляции базового модуля ZZ (или модуля ZZ-SWAP) в конкретной физической реализации.

RSA	**Кубиты**	**Kn глубина**	**2DSL глубина**	**LNN глубина**
RSA-128	37	113	121	150
RSA-256	64	194	204	258
RSA-512	114	344	357	458
RSA-1024	205	617	633	822
RSA-2048	372	1118	1139	1490

Мы также проанализировали масштаб RSA-чисел, а именно предельный размер (touch-size), которого могут достичь современные квантовые вычислительные устройства при некоторых идеальных условиях, при которых все заявленные кубиты считаются относительно идеальными или обладающими высокой достоверностью. Результаты приведены в соответствии с топологией связности кубитов квантовых устройств с использованием алгоритма SQIF. Рассматриваемые квантовые процессоры включают: Sycamore, Eagle, Aspen-M, Zuchongzhi2, Tianmu-1, а также ионные ловушки, разработанные в Мэриленде и компанией IonQ. Все эти устройства либо находятся в публичном доступе, либо доступны через облачные квантовые платформы. Кроме того, мы указываем минимальную необходимую глубину схемы (“depth-least”) для этих устройств, при которой можно попытаться факторизовать RSA-числа предельного размера. Эта глубина оценивается по глубине однослойной схемы QAOA.В частности, если квантовый процессор имеет двумерную решётчатую топологию, глубина схемы рассчитывается по модели 2DSL. Для топологий типа “прочие” глубина рассчитывается по модели LNN. Подробные результаты приведены в Таблице З.11.

Из Таблицы З.11 видно, что предельный размер для устройств NISQ уже близок к реальным RSA-числам. Например, 127-кубитная машина `ibm-eagle` имеет предельный размер 581, а минимально необходимая глубина схемы составляет 510. Это означает, что если все кубиты функционируют стабильно и обеспечивают достаточную точность после 510 уровней схемы, устройство может быть использовано для попытки факторизации RSA-581.

Тем не менее, предельный размер — это идеализированная оценка. На практике алгоритм QAOA обычно требует более чем одного слоя, что увеличивает глубину схемы. Кроме того, наличие квантового ускорения по-прежнему остаётся неизвестным, и квантовый взлом RSA пока остаётся далёкой целью.

Таблица З.11 – Предельный размер RSA-чисел для некоторых известных квантовых устройств. Результаты приведены на основе топологии связности кубитов и набора базовых логических затворов соответствующих квантовых устройств, с использованием предложенного нами алгоритма. Последний столбец указывает минимальное условие по глубине схемы, которое должно быть выполнено для возможности факторизации. Для систем с топологией типа “прочие” глубина схемы рассчитывается по модели LNN.

Система	**Устройство**	**Кубиты**	**Топология**	**Предельный размер**	**Мин. глубина**
Сверхпроводящие кубиты	Sycamore	53	2DSL	201	170
	Eagle	127	прочие	581	510
	Aspen-M	80	прочие	334	322
	Zuchongzhi2	66	2DSL	264	210
	Tianmu-1	36	2DSL	124	118
Ионные ловушки	Maryland	40	K_n	142	122
	IonQ	79	K_n	329	239