УЧЕНЫЕ ЗАПИСКИ КАЗАНСКОГО УНИВЕРСИТЕТА.
СЕРИЯ ФИЗИКО-МАТЕМАТИЧЕСКИЕ НАУКИ

2020, Т. 162, кн. 3
С. 241–258

ISSN 2541-7746 (Print)
ISSN 2500-2198 (Online)

УДК 519.7

doi: 10.26907/2541-7746.2020.3.241-258

КВАНТОВЫЙ ПОИСК ЗАДАННОЙ ПОДСТРОКИ В ТЕКСТЕ НА ОСНОВЕ ТЕХНИКИ ХЕШИРОВАНИЯ

М.Ф. Аблаев[1,2], *Н.М. Салихова*[2]

[1]*Казанский физико-технический институт им. Е.К. Завойского ФИЦ Казанский научный центр РАН, г. Казань, 420029, Россия*

[2]*Казанский (Приволжский) федеральный университет, г. Казань, 420008, Россия*

Аннотация

Рассмотрена задача поиска в тексте заданной подстроки. Известно, что классические алгоритмы решают эту задачу за линейное от длины текста и заданного шаблона время. Квантовые алгоритмы позволяют ускорить поиск в «корень квадратный раз».

В работе предложен квантовый алгоритм, решающий задачу поиска а) с высокой вероятностью получения правильного результата, б) с таким же ускорением («корень квадратный раз») по сравнению с классическим, но в) гораздо более экономный по памяти (по числу используемых кубит) по сравнению с ранее известными квантовыми алгоритмами.

Продемонстрировано применение универсального хеширования и техники квантового хеширования для задачи поиска в тексте заданной подстроки.

Ключевые слова: квантовые алгоритмы, поиск строки, квантовый поиск

Введение

Современные исследования и разработки квантовых алгоритмов ведутся (*уже более* или, в зависимости от степени вовлеченности читателя, *еще только*) три десятилетия [1]. На сегодняшний день известные квантовые алгоритмы можно отнести в один из четырех классов: а) алгоритмы типа алгоритмов проблемы скрытых подгрупп (Hidden Subgroup Problem), включающий известный алгоритм факторизации Шора, б) алгоритмы поиска [2–4], в) алгоритмы на основе квантового блуждания (Quantum Walk) [5] и г) системы квантового моделирования (симуляций), которые собственно и были первоначальной мотивацией, из которой исходил Ричард Фейнман, когда начал говорить о квантовых моделях вычислений в 80-х годах XX в.

Квантовые алгоритмы применяются для решения задач линейной алгебры, широко используемых в машинном обучении, таких как вычисление обратной матрицы, нахождение собственных чисел и собственных векторов матрицы. Для квантовых компьютеров активно разрабатываются аналоги классических алгоритмов машинного обучения. Для моделей с большим количеством весов или таких, для которых требуется перебор экспоненциально растущего числа комбинаций, использование квантовых компьютеров позволяет значительно сократить время вычисления. Обзор некоторых результатов в области квантовых подходов к классификации и поиску информации представлен нами работах [6, 7]. Задача поиска вхождений заданной подстроки в тексте – одна из основных задач поиска информации. Она возникает в широком спектре приложений: в текстовых редакторах, в работе поисковых роботов, спам-фильтров, в биоинформатике и др. За последние несколько

десятилетий для решения такой задачи поиска разработано большое число алгоритмов (детерминированных и вероятностных), а в последние два десятилетия появились и квантовые алгоритмы.

Постановка задачи. Дана двоичная последовательность (строка) $string$ длины N: $string = b_1 \dots b_N$. Дана двоичная строка w длины m, $m < N$. Требуется найти индекс вхождения подстроки w в строке $string$. А именно, требуется найти индекс k такой, что для $string$ и w выполняется $w = b_k \dots b_{k+m-1}$.

Будем использовать следующие обозначения. Через $T(string)$ обозначим *последовательность* всех слов длины m текста $string$. А именно, для $n = N + 1 - m$ последовательность $T(string)$ составлена из всех подстрок w_k длины m строки $string$:

$$T(string) = \{w_0, \cdots, w_{n-1}\},$$

где $w_k = b_{k+1} \cdots b_{k+m}$ для $0 \le k \le n-1$.

Известные результаты. Алгоритм Кнута – Морриса – Пратта [8] 70-х годов XX в. решает задачу за линейное время $O(m+n)$.

В начале 2000-х годов представлен квантовый алгоритм [9] поиска заданной подстроки в тексте. Он относится ко второй группе алгоритмов и позволяет получить квадратичное ускорение поиска. Он возвращает один из индексов вхождения искомой подстроки. Вероятность получения правильного ответа строго больше $1/2$. Сложность запросов (эта мера сложности иногда ассоциируется с временной сложностью) такого алгоритма составляет $O(\sqrt{n}\log\sqrt{\frac{n}{m}}\log m + \sqrt{m}\log^2 m)$. Авторы [9] не оценивают сложность по памяти (количество используемых кубит) своего алгоритма. Наш анализ показывает, что алгоритму [9] требуется $O(\log n + m)$ кубит для работы.

Результаты работы. Мы строим алгоритмы для случая, когда заранее точно известно, что искомая подстрока встречается в тексте ровно один раз. Такой случай рассмотрен Л. Гровером в работе [2], положившей начало исследованиям в области квантового поиска. Мы предлагаем квантовый алгоритм поиска $\mathcal{A}$, который проводит поиск подстроки в тексте

- с большой вероятностью получения правильного ответа,
- более экономичен (по сравнению с алгоритмом [9]) по времени, а именно требует $O(\sqrt{n}\log m)$ шагов работы алгоритма,
- позволяет экспоненциально (по сравнению с алгоритмом [9]) сократить количество кубит относительно параметра m – длины шаблона, а именно алгоритму достаточно $O(\log n + \log m)$ кубит для своей работы.

Ниже мы продемонстрируем потенциальные возможности универсального хеширования и техники квантового хеширования для построения алгоритмов квантового поиска подстроки в тексте.

1. Квантовый алгоритм запросов, сложность и необходимые сведения

В настоящей работе мы используем систему обозначений, определяемых подробно в работах [6, 7]. Операции, применяемые к квантовым s-кубитным состояниям $|\psi\rangle \in (\mathcal{H}^2)^{\otimes s}$, математически выражаются с помощью унитарных операторов

$$|\psi'\rangle = U|\psi\rangle, \tag{1}$$

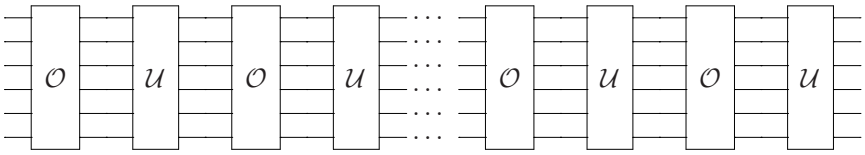
где U – $2^s \times 2^s$ унитарная матрица.

1.1. Модель вычисления. Мы определяем алгоритм на основе известной в теории квантовых вычислений модели – квантового алгоритма запросов (Quantum Qurey Complexity Model (QQCM)). Мы определяем квантовый алгоритм запросов в самом общем виде, следуя работе [10].

Квантовый алгоритм запросов $\mathcal{A}$ на s кубитах ($s = \text{size}(\mathcal{A})$) для вычисления значения дискретной функции $g(X)$ определяется следующим образом. Пусть $|\psi_{\text{start}}\rangle$ - начальное s-кубитное состояние. Процедура вычисления функции задается последовательностью операторов

$$\mathcal{U},\mathcal{O},\mathcal{U},\ldots,\mathcal{O},\mathcal{U},$$

$\dim(A) \times \dim(A)$ унитарных матриц ($\dim(A) = 2^s$).



Оператор $\mathcal{U}$ не зависит от X. Оператор $\mathcal{O}$ зависит от входного X, ($\mathcal{O} = \mathcal{O}(X)$). Оператор $\mathcal{O}$ в литературе называют «оракул». Применение оракула называют запросом алгоритма $\mathcal{A}$ к исходным данным X.

Алгоритм состоит в применении последовательности операторов $\mathcal{O},\mathcal{U},\ldots,\mathcal{O},\mathcal{U}$ к $|\psi_{\text{start}}\rangle$. Алгоритм вычисляет дискретную функцию $g(X)$, если в процессе вычисления на входном значении σ начальное состояние $|\psi_{\text{start}}\rangle$ трансформируется в конечное состояние

$$|\psi(g(\sigma))\rangle = \mathcal{U}\mathcal{O}\cdots\mathcal{U}\mathcal{O}|\psi_{\text{start}}\rangle,$$

которое позволяет извлечь значение $g(\sigma)$ в результате измерения состояния $|\psi(g(\sigma))\rangle$.

1.2. Характеристики алгоритма $\mathcal{A}$. Характеристиками квантового алгоритма запросов являются вероятность ошибки его работы, число запросов к анализируемым данным и используемая память.

Вероятность ошибки. Пусть $\epsilon \in (0,1)$. Будем говорить, что квантовый алгоритм запросов $\mathcal{A}$ вычисляет функцию $g(X)$ с вероятностью ошибки не более ϵ, если для каждого значения σ переменных X вероятность извлечения из состояния $|\psi(g(\sigma))\rangle$ значения, отличного от $g(\sigma)$, не превосходит ϵ.

Сложность запросов. Число $Q(\mathcal{A})$ запросов (число применений оракула) является «запросной» мерой сложности квантового алгоритма $\mathcal{A}$.

Отметим, что в работе [10] один запрос к оракулу – это тестирование одной переменной (одного бита). Соответственно, в нашей работе один запрос к оракулу реализуется в процессе тестировании целой строки w для алгоритма $\mathcal{A}1$ (хеш-значения w в алгоритмах $\mathcal{A}2$, $\mathcal{A}3$).

Сложность по памяти. Число $S(\mathcal{A}) = s$ используемых кубит является мерой сложности памяти квантового алгоритма $\mathcal{A}$.

1.3. Квантовый алгоритм поиска. Квантовый алгоритм поиска в неупорядоченной базе данных из n элементов, в которой присутствует ровно один элемент, нас интересующий, (алгоритм Гровера) – частный случай группы алгоритмов запросов. Следующая схема алгоритма лежит в основе многих обобщений алгоритмов поиска.

A. Инициализировать квантовую систему из $O(\log n)$ кубитов в состояние $|\psi_{\text{start}}\rangle$, содержащее всю информацию о базе данных. Состояние $|\psi_{\text{start}}\rangle$ конструируется так, чтобы каждое из $2^{O(\log n)}$ базисных квантовых состояний представляло нужную информацию об n элементах базы данных.

B. Над состоянием $|\psi_{\text{start}}\rangle$ выполняются следующие $O(\sqrt{n})$ «макро»-шагов:

• применяется оракул $\mathcal{O}$, который распознает состояние, интересующее нас, и умножает его амплитуду на -1;

• применяется оператор, выполняющий инверсию по среднему значению по всем амплитудам.

«Макро»-шаг 2 описывает ключевую операцию квантового поиска. Каждый такой шаг увеличивает амплитуду состояния, представляющего нужную информацию. Число $O(\sqrt{n})$ таких шагов максимизирует амплитуду, а следовательно, и вероятность извлечения нужной информации из результирующего состояния $|\psi_{\text{final}}\rangle$.

1.4. Необходимые базисные операции кубитов. Потенциальные преимущества квантовых алгоритмов, на которых базируются результаты настоящей работы, заключаются в возможности реализации квантовых операторов размерности $2^s \times 2^s$ базисными операциями на основе небольшого числа порядка $O(s)$ кубит и за небольшое число шагов вычислений квантового вычислителя (несложной схемой, реализуемой базисными элементами).

Представим операторы $I,\ X,\ Z,\ H$, которые будут использоваться нами:

• I – оператор идентичности

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix};$$

• X – оператор отрицания. Он меняет состояние кубита с $|0\rangle$ на $|1\rangle$ и наоборот

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}; \tag{2}$$

• Z – оператор изменения знака амплитуды

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}; \tag{3}$$

• H – оператор Адамара

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{4}$$

Отметим, что вектор из гильбертова линейного пространства обозначают как кет-вектор $|x\rangle$, а соответствующий ему вектор из сопряженного пространства – как бра-вектор $\langle x|$.

1.5. Теорема о распределении простых чисел. Простых чисел много, а именно существует примерно $\dfrac{n}{\ln n}$ простых чисел, меньших числа n. k-е по счету простое число $p_k \approx k \ln k,\ k \to \infty$.

2. Алгоритмы поиска индекса вхождения подстроки в тексте

В данном разделе мы представим три квантовых алгоритма поиска подстроки в тексте. Задача рассматривается в простейшей постановке. А именно, предполагается, что искомая подстрока w гарантированно встречается в тексте $string$ и

ровно один раз. Обобщения на более общие случаи обсуждаются в заключительном разделе статьи.

2.1. Исходные данные и результат. Исходными данными для всех трех алгоритмов являются текст $string$ и искомая подстрока w. Алгоритм строится в предположении, что в тексте $string$ искомая подстрока w присутствует и присутствует ровно один раз. Результатом работы алгоритмов является номер k вхождения w в текст $string$.

Для упрощения выкладок всюду далее будем считать, что число n (число подслов в тексте $string$) – степень 2.

Алгоритм $\mathcal{A}1$. Первый алгоритм излагается и анализируется достаточно подробно. $\mathcal{A}1$ является квантовой процедурой, используемой двумя последующими алгоритмами, и в то же время представляет самостоятельный интерес.

2.2. Операторное задание.

1. По строке $string$ готовится начальное $(\log n + m + 1)$ кубитное состояние $|string\rangle$ на основе последовательности $T(string)$

$$|string\rangle = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} |k\rangle \otimes |w_k\rangle \otimes |1\rangle.$$

2. К состоянию $|string\rangle$ применяются $\frac{\pi}{4}\sqrt{n}$ раз следующие два макрошага, описываемые ниже в операторном виде. В литературе эти два макрошага часто называют «итерация Гровера» [2].

Операция изменения фазы состояния, представляющего информацию о w_k, для которого выполняется $w_k = w$. Для двоичной последовательности $w \in \{0,1\}^m$ определяем булеву функцию $f_w : \{0,1\}^m \to \{0,1\}$ условием: $f_w(x) = 1$ тогда и только тогда, когда $x = w$.

Обозначим через $|x\rangle$ базисное состояние, соответствующее элементу x, являющимся одним из w_k. В данном случае $|x\rangle$ – это m кубитное базисное состояние. Оракул O_{f_w} (представляется, что термин *оракульное преобразование* более точно отражает ситуацию, если уж следовать традиции квантового поиска и использовать термин *оракул*) задается последовательностью следующих действий:

а) применение оператора H на вспомогательном $(\log n + m + 1)$-м кубите $|1\rangle$:

$$|x\rangle \otimes |1\rangle \xrightarrow{H} |x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle);$$

б) применение оператора U_{f_w} на последних $m+1$ кубитах:

$$|x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \xrightarrow{U_{f_w}} \frac{1}{\sqrt{2}}|x\rangle \otimes (|0 \oplus f_w(x)\rangle - |1 \oplus f_w(x)\rangle) =$$
$$= (-1)^{f_w(x)}|x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle);$$

в) применение оператора H на вспомогательном $(\log n + m + 1)$-м кубите $|1\rangle$:

$$(-1)^{f_w(x)}|x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \xrightarrow{H} (-1)^{f_w(x)}|x\rangle \otimes |1\rangle.$$

Отметим, что вспомогательный кубит в конце восстанавливает свое значение (повторным применением преобразования Адомара) в состояние $|1\rangle$, способствуя изменению фазы композиции $|x\rangle \otimes |1\rangle$ в зависимости от значения $f_w(x)$.

Операция инверсии. $\mathcal{D} = 2\mathcal{R}$ - $I^{\otimes \log n}$ – оператор, применяемый на первых n кубитах. Оператор $\mathcal{R}$ задается в матричном виде:

$$\mathcal{R} = \frac{1}{\log n} \begin{pmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \dots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix}.$$

3. *Получение результата вычисления.* После проведения описанных выше макрошагов (шага операции изменения фазы состояния, представляющего информацию о w_k, для которого выполняется $w_k = w$ и шага операции инверсии) $\frac{\pi}{4}\sqrt{n}$ раз измеряем первые $\log n$ кубит в классическом базисе. Ответ: результат измерения.

2.3. Меры сложности и вероятность ошибки алгоритма. Операторное задание. Через $Q^{\mathcal{A}}(string, w)$ обозначим число применений алгоритмом $\mathcal{A}$ оператора O_{f_w} (число запросов алгоритма к анализируемому тексту) при решении задачи поиска подстроки w в тексте $string$ с учетом сложности реализации самого оператора запроса O_{f_w}.

Через $Q^{\mathcal{A}}(n, m)$ обозначим максимум среди чисел $Q^{\mathcal{A}}(string, w)$ по всем $string$ и w с параметрами $N = |string|, m = |w|, n = N - m + 1$.

Через $S^{\mathcal{A}}(string, w)$ обозначим число кубит, используемых алгоритмом $\mathcal{A}$ для решения задачи поиска подстроки w в строке $string$.

Через $S^{\mathcal{A}}(n, m)$ обозначим максимум среди чисел $S^{\mathcal{A}}(string, w)$ по всем $string$ и w с параметрами $N = |string|, m = |w|, n = N - m + 1$.

Через $Er^{\mathcal{A}}(string, w)$ обозначим вероятность ошибки алгоритма, то есть $Er^{\mathcal{A}}(string, w)$ – это вероятность следующего события: алгоритм $\mathcal{A}$ в результате решения задачи поиска подстроки w в тексте $string$ выдает номер k позиции в тексте $string$ такой, что $w_k \neq w$.

Через $Er^{\mathcal{A}}(n, m)$ обозначим максимум среди чисел $Er^{\mathcal{A}}(string, w)$ по всем $string$ и w с параметрами $N = |string|, m = |w|, n = N - m + 1$.

Пусть $\epsilon \in (0, 1)$. Будем говорить, что квантовый алгоритм $\mathcal{A}$ для задачи поиска подстроки w в строке $string$ решает задачу с вероятностью ошибки не более ϵ, если

$$Er^{\mathcal{A}}(n, m) \leq \epsilon.$$

Теорема 1. *Для $\epsilon = 1/n$ выполняется*

$$\begin{aligned} Er^{\mathcal{A}1}(n, m) &\leq \epsilon, \\ Q^{\mathcal{A}1}(n, m) &= O(m\sqrt{n}), \\ S^{\mathcal{A}1}(n, m) &= \log n + m + 1. \end{aligned}$$

Доказательство. Для кодировки номера подстроки необходимо $\log n$ кубит, m кубит – для кодирования соответствующих подстрок w_k и 1 вспомогательный кубит. Таким образом, $S_{\epsilon}^{\mathcal{A}1}(n, m) = \log n + m + 1$.

Рассмотрим динамику изменения амплитуд при применении алгоритма $\mathcal{A}1$ к начальному состоянию $|string\rangle$. Положим $|string_0\rangle = |string\rangle$. Обозначим через α_0 исходную амплитуду при искомом базисном состоянии $|k\rangle|w_k\rangle|1\rangle$ таком, что $w_k = w$, а через β_0 амплитуды всех остальных базисных состояний состояний начального состояния $|string_0\rangle$, то есть $\alpha_0 = 1/\sqrt{n} = \beta_0 = 1/\sqrt{n}$ и $\alpha_0^2 + (n-1)\beta_0^2 = 1$. В силу введенных обозначений состояние $|string_0\rangle$ представимо в следующем

виде:

$$|string_0\rangle = \alpha_0 \sum_{k:w_k=w} |k\rangle \otimes |w_k\rangle \otimes |1\rangle + \beta_0 \sum_{k:w_k\neq w} |k\rangle \otimes |w_k\rangle \otimes |1\rangle.$$

После применения к начальному состоянию $|string_0\rangle$ последовательно j раз двух макрошагов: 1) операции изменения фазы состояния, представляющего информацию о w_k, для которого выполняется $w_k = w$, и 2) операции инверсии (называемых в литературе итерацией Гровера) – амплитуды $(j+1)$-го состояния $|string_{j+1}\rangle$ будут выражаться формулами (см., например, работу [11] для подробного технического обоснования воздействия операторов 1) и 2) на состояния $|string_j\rangle$):

$$\alpha_{j+1} = \frac{n-2}{n}\,\alpha_j + \frac{2(n-1)}{n}\,\beta_j, \quad \beta_{j+1} = \frac{n-2}{n}\,\beta_j - \frac{2}{n}\,\alpha_j.$$

Имеем

$$|string_{j+1}\rangle = \alpha_{j+1} \sum_{k:w_k=w} |k\rangle \otimes |w_k\rangle \otimes |1\rangle + \beta_{j+1} \sum_{k:w_k\neq w} |k\rangle \otimes |w_k\rangle \otimes |1\rangle,$$

где (напомним, что рассматриваем случай, когда k, для которого $w_k = w$, является единственным номером):

$$\alpha_{j+1}^2 + (n-1)\beta_{j+1}^2 = 1.$$

Поэтому величины α и β можно задать как

$$\alpha_j = \sin\{(2j+1)\theta\}, \quad \beta_j = \frac{1}{\sqrt{n-1}} \cos\{(2j+1)\theta\}.$$

Далее $\alpha_r = 1$, если $(2r+1)\theta = \pi/2$. На основе этих соображений определяется оптимальное число r итераций алгоритма поиска $r = (\pi - 2\theta)/4\theta$.

В работе [11] показано, что вероятность получения ошибочного результата не превосходит $1/n$, если выполнить последовательно $[\pi/(4\theta)]$ итераций Гровера. Если n достаточно большое число, то $\theta \approx \sin\theta = 1/\sqrt{n}$, тогда

$$r = \frac{\pi}{4}\sqrt{n}.$$

Так как в алгоритме одно обращение к оракулу – это тестирование целой подстроки, состоящей из m бит, то итоговое значение $Q^{\mathcal{A}1}(n,m) = O(m\sqrt{n})$. □

2.4. Задание алгоритма в виде квантовой схемы. В общем виде детальная схема оракула представлена на рис. 1. Операция контролируемого поворота фазы последнего кубита U_{f_w} отражена совокупностью двух прямоугольников W и Z. Внутреннее устройство прямоугольника W определяется искомым подсловом w.

Пример схемы оракула в случае, когда $w = 01\ldots0$, представлена на рис. 2. Каждому биту, кодирующему подстроку w, ставится в соответствие кубит из предпоследних m кубит регистра. На кубитах, соответствующих нулям из двоичной последовательности, кодирующей подстроку w, применяется гейт X два раза: до операции контролируемого изменения знака амплитуды на вспомогательном кубите, где предпоследние m кубит являются контролирующими, и после. Схема оператора $\mathcal{D}$ представлена на рис. 3. Схема итерации Гровера представлена на рис. 4. Итерация Гровера G обведена в пунктирную рамку. На общей схеме алгоритма, представленной на рис. 5, один пунктирный прямоугольник обозначает

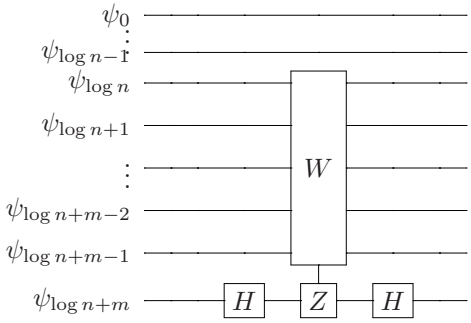


Рис. 1. Квантовая схема оракула в алгоритме поиска подстроки в общем виде

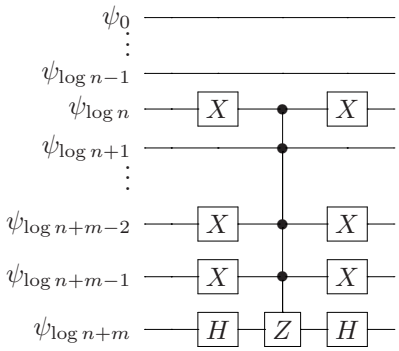


Рис. 2. Квантовая схема оракула в алгоритме поиска подстроки $01\ldots0$

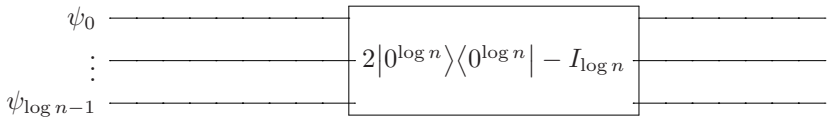


Рис. 3. Схема оператора $\mathcal{D}$

одну итерацию Гровера. Их необходимо повторить $\frac{\pi}{4}\sqrt{n}$ раз. После этого следует измерение первых $\log n$ кубит.

Количество запросов,выполняемых алгоритмом, равно $O(m\sqrt{n})$.

Результат работы алгоритма – число k такое, что $w = w_k$.

2.5. Схемная сложность. Схемную сложность $Q\mathrm{Size}^{\mathcal{A}1}(n,m)$ квантового алгоритма $\mathcal{A}1$ будем определять как число базисных операций, необходимых для реализации квантовой схемы, реализующей алгоритм $\mathcal{A}1$.

Теорема 2. $Q\mathrm{Size}^{\mathcal{A}1}(n,m) = O(\sqrt{n}(m+\log n))$.

Доказательство. Действительно из описания схемы имеем, что

$$Q\mathrm{Size}^{\mathcal{A}1}(n,m) = O(\sqrt{n}\,\mathrm{Size}(G)),$$

где $\mathrm{Size}(G)$ – это схемная сложность реализации итерации Гровера для данного алгоритма $\mathcal{A}1$. Схемная сложность $\mathrm{Size}(G)$ итерации Гровера для $\mathcal{A}1$ складывается

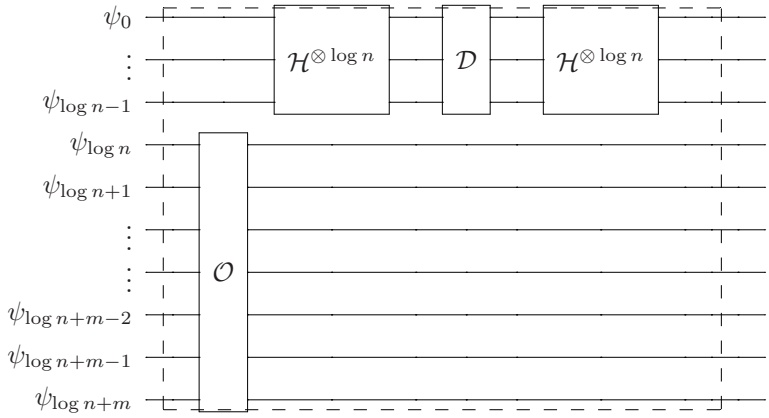


Рис. 4. Квантовая схема итерации алгоритма Гровера

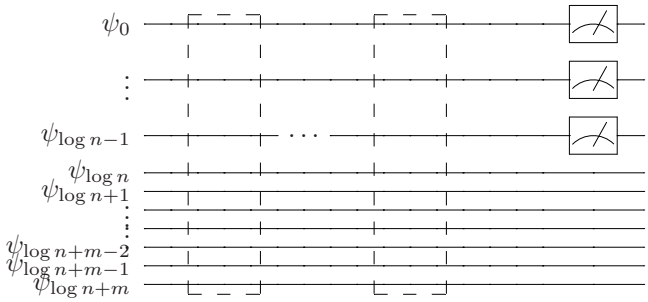


Рис. 5. Квантовая схема алгоритма поиска.

из схемной сложности $\mathrm{Size}(\mathcal{O}_f)$ схемы $\mathcal{O}$ и из схемной сложности $\mathrm{Size}(\mathcal{D}) + 2\log n$ схемы $\mathcal{D}$:

$$\mathrm{Size}(G) = \mathrm{Size}(\mathcal{O}_f) + \mathrm{Size}(\mathcal{D}) + 2\log n.$$

В силу описания реализации оператора $\mathcal{O}_f$ имеем $\mathrm{Size}(\mathcal{O}_f) \le 2m + 3$. Оператор диффузии $\mathcal{D}$ является стандартным. Его реализация рассмотрена в ряде работ (см., например, работу [12]). Сложность схемы для $\mathcal{D}$ оценивается величиной $\mathrm{Size}(\mathcal{D}) \le 4\log n$. □

Алгоритм $\mathcal{A}2$. Двоичную строку (подстроку) w длины m будем считать также числом, $0 \le w \le 2^m - 1$. Обозначим через $\mathrm{Set}(string)$ *множество* (словарь текста $string$) всех различных подслов длины m в строке $string$. Напомним, что $T(string)$ – это *последовательность* слов длины m, образованная из $string$. Понятно, что $|\mathrm{Set}(string)| \le |T(string)|$

В данном разделе $P = \{p_1, \ldots, p_d\}$ будет обозначать множество первых d простых чисел, где $d = d(n, m) = cnm$, для целого $c \ge 3$.

Дадим описание алгоритма $\mathcal{A}2$.

Первый этап алгоритма – классический: равновероятностно выбирается простое число p из множества P.

Второй этап (*подготовка квантового состояния*): для числа $v \in \{0, \ldots, 2^m - 1\}$ и $p \in P$ обозначим через $r(v)_p$ – остаток от деления v/p, то есть $v = c_1 p + r(v)_p$, для некоторого c_1. Через

$$T_p(string) = \{r(w_0)_p, \ldots, r(w_{n-1})_p\}$$

обозначим последовательность слов (здесь остатки $r(w_j)_p$ уже рассматриваются как двоичные слова), образованную из $string$. По $T_p(string)$ порождается квантовое состояние:

$$|string, p\rangle = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} |k\rangle \otimes |r(w_k)_p\rangle \otimes |1\rangle.$$

Третий этап (*квантовый*) алгоритма: к состоянию $|string, p\rangle$ и искомому слову w применяется алгоритм $\mathcal{A}1$. В результате измерения первых $\log n$ кубит будет получен номер $j \in \{0, \ldots, n-1\}$, который выдается в качестве ответа и объявляется искомым номером слова w_j такого, что $w_j = w$.

Комментарий. Алгоритм $\mathcal{A}1$ в данном случае играет роль подпрограммы для алгоритма $\mathcal{A}2$, применяемой для текста $string$ и искомой последовательности w при выбранном параметре (простом числе) $p \in P$. В дальнейших выкладках результат k (номер слова w_k, для которого предположительно выполняется $r(w_k)_p = r(w)_p$) работы алгоритма $\mathcal{A}1$ будем обозначать как $\mathcal{A}1(string, w, p)$, то есть $\mathcal{A}1$ реализует отображение

$$\mathcal{A}1 : (string, w, p) \longmapsto k.$$

Теорема 3. *Для произвольного целого $c \geq 3$ выполняется*

$$Er^{\mathcal{A}2}(string, w)) \leq \frac{1}{c} + \frac{1}{n},$$
$$Q^{\mathcal{A}2}(n, m) = O(\sqrt{n} \log m),$$
$$S^{\mathcal{A}2}(n, m) \leq O(\log n + \log m).$$

Доказательство теоремы 3 приводится ниже.

2.6. Доказательство теоремы 3. Для пары различных чисел $w_1, w_2 \in \{0, \ldots 2^m - 1\}$ обозначим через P_{w_1,w_2} множество простых $p \in P$ таких, что $w_1 \equiv w_2 \pmod p$.

При доказательстве теоремы используется следующая

Лемма 1. *Для произвольной пары различных чисел $w_1, w_2 \in \{0, \ldots, 2^m - 1\}$ выполняется*

$$|P_{w_1,w_2}| \leq m.$$

Доказательство. Для простых $p \in P$ имеем, что если $w_1 \equiv w_2 \pmod p$, то их разность $a = |w_1 - w_2| \in \{0, \ldots, 2^m - 1\}$ кратна p. Значит,

$$a < 2^m < p_1 p_2 \cdots p_m,$$

то есть число простых, делящих a, не превосходит m. Следовательно, для различных w_1 и w_2 количество простых чисел из P, которые будут давать одинаковые остатки при делении w_1 и w_2 на них, ограничено сверху величиной m. □

Далее докажем утверждения теоремы 3.

Оценка вероятности $Er^{\mathcal{A}2}(string, w)$ ошибки основывается на следующих соображениях. Для $string$ и искомого слова w разобьем множество P на хорошее P_{good} и плохое P_{bad}. Простое $p \in P$ будем считать хорошим для $string$ и w, если $r(w)_p \neq r(w_j)_p$ для всех $w_j \in T(string)$ таких, что $w \neq w_j$. Тогда вероятность ошибки алгоритма $\mathcal{A}2$ можно оценить сверху как вероятность ошибки алгоритма (процедуры) $\mathcal{A}1$ при выпадении хорошего $p \in P$ плюс вероятность выпадения плохого $p \in P$. При этом при выпадении плохого p возможен случай получения

верного результата при применении процедуры $\mathcal{A}1$. Однако, считая все продолжения работы алгоритма $\mathcal{A}1$ ошибочными при плохих p, мы только увеличиваем величину вероятности ошибки.

Рассмотрим более формально. Для множества $\mathrm{Set}(string)$ и искомой последовательности w положим

$$P_{\mathrm{bad}} = \bigcup_{v \in \mathrm{Set}(string)} P_{w,v}, \quad P_{\mathrm{good}} = P \backslash P_{\mathrm{bad}}.$$

Тогда вероятность $Er^{\mathcal{A}2}(string, w)$ оценивается так:

$$Er^{\mathcal{A}2}(string, w) \le \frac{|P_{\mathrm{bad}}|}{|P|} + \frac{|P_{\mathrm{good}}|}{|P|} \max_{p \in P_{\mathrm{good}}} \{Er^{\mathcal{A}1}(string, w, p)\} \le$$
$$\le \frac{|P_{\mathrm{bad}}|}{|P|} + \max_{p \in P_{\mathrm{good}}} \{Er^{\mathcal{A}1}(string, w, p)\}.$$

В силу леммы 1 имеем $|P_{\mathrm{bad}}| \le mn$. Комбинация двух предыдущих неравенств, верхняя оценка (теорема 1) для $Er^{\mathcal{A}1}$ и выбор множества P дают окончательную оценку

$$Er^{\mathcal{A}2}(string, w) \le \frac{nm}{cnm} + \frac{1}{n} = \frac{1}{c} + \frac{1}{n}.$$

Оценка величины запросной сложности $Q^{\mathcal{A}2}(n, m)$ следует из доказательства запросной сложности алгоритма $\mathcal{A}1$. Отличие запросной сложности алгоритма $\mathcal{A}2$ от запросной сложности алгоритма $\mathcal{A}1$ заключается в том, что вместо числа m в алгоритме $\mathcal{A}2$ используется число $\log m$ в силу сокращения количества используемых кубит для кодировки подстроки.

Далее оценим величину используемого количества $S^{\mathcal{A}2}(n, m)$ кубит.

Для кодировки номера подслова в строке $string$ используется $\log n = \log(N - m + 1)$ кубит. Количество кубит, необходимых для кодировки остатка от деления w на p, зависит от величины p. Самое большое значение, которое может принять p – это простое число p_{cnm} под номером cnm. По свойству простых чисел имеем

$$p_{cnm} \le c_1 \frac{cnm}{\ln cnm} \ln \frac{cnm}{\ln cnm}$$

или

$$p_{cnm} \le cnm \ln cnm.$$

Количество кубит, необходимое для кодирования такого числа, не превышает величины

$$\log(cnm \ln cnm) = \log c + \log n + \log m + \log(\ln cnm).$$

Таким образом, общее количество кубит оценивается сверху величиной $O(\log n + \log m)$.

Алгоритм $\mathcal{A}3$. Третий алгоритм – «более квантовый, чем второй». В третьем алгоритме классический выбор простого $p \in P$, при котором вызывается процедура $\mathcal{A}1(string, w, p)$, заменяется на квантовый способ выбора такого p. За это мы «платим» привлечением дополнительного числа кубит и усложнением (слегка) начального состояния алгоритма и самого алгоритма. Поэтому более правильно было бы назвать этот алгоритм не $\mathcal{A}3$, а $\mathcal{A}2+$. Но мы сохраним обозначение $\mathcal{A}3$ и порядковый номер «третий».

Пусть двоичная строка (подстрока) $w = w_1 \dots w_m$ длины m рассматривается как число, $0 \le w \le 2^m - 1$. Напомним, что $P = \{p_1, \dots, p_d\}$ – множество первых d простых чисел. $d = d(n, m) = cnm$, $c \ge 3$ – целое.

Первый этап (подготовка квантового состояния).

Напомним, что через $r(w)_p$ обозначается остаток от деления w/p, то есть $w = c_1 p + r(w)_p$ для константы $c_1 \geq 0$. Порождается квантовое состояние:

$$\frac{1}{\sqrt{d}} \sum_{l=0}^{d-1} |l\rangle \left|string_{p_{l+1}}\right\rangle,$$

где

$$\left|string_{p_{l+1}}\right\rangle = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} |k\rangle \otimes \left|r(w_k)_{p_{l+1}}\right\rangle \otimes |1\rangle.$$

Второй этап (выбор простого числа).

Производится измерение первых $\log d$ кубит, то есть производится выбор номера $l \in \{0, \ldots, d-1\}$, тем самым выбирается простое число $p_{l+1} \in P$.

Третий этап (квантовый).

К состоянию $\left|string_{p_{l+1}}\right\rangle$ применяется алгоритм $\mathcal{A}1$. В результате измерения первых $\log n$ кубит будет получен номер $j \in \{0, \ldots, n-1\}$, такой, что $r(w)_{p_{l+1}} = r(w_j)_{p_{l+1}}$.

Так как $r(w)_{p_{l+1}} = w \bmod p_{l+1}$, $r(w_j)_{p_{l+1}} = w_j \bmod p_{l+1}$, то номер j и является искомым номером вхождения подстроки w в строке $string$.

Теорема 4. *Для произвольного целого $c \geq 3$ выполняется*

$$Er^{\mathcal{A}3}(string, w)) \leq \frac{1}{c} + \frac{1}{n},$$
$$Q^{\mathcal{A}3}(n, m) = O(\sqrt{n} \log m),$$
$$S^{\mathcal{A}3}(n, m) \leq O(\log n + \log m).$$

Действительно, ошибка равна $Er^{\mathcal{A}3}(string, w))$, сложность $Q^{\mathcal{A}3}(n, m)$ запросов алгоритма $\mathcal{A}3$ совпадает со сложностью алгоритма $\mathcal{A}2$. К величине $S^{\mathcal{A}3}(n, m)$ сложности по памяти добавляется $\log d = \log cnm$ кубит. Последнее совпадает с оценкой теоремы 3 с точностью до мультипликативной константы.

Из теоремы 2 и описания алгоритмов $\mathcal{A}2$ и $\mathcal{A}3$ вытекают следующие оценки на схемную сложность реализации алгоритмов $\mathcal{A}2$ и $\mathcal{A}3$.

Теорема 5.

$$QSize^{\mathcal{A}2}(n, m) = O(\sqrt{n} \log (mn))$$

и

$$QSize^{\mathcal{A}3}(n, m) = O(\sqrt{n} \log (mn)).$$

Доказательство. Действительно, алгоритмы $\mathcal{A}2$ и $\mathcal{A}3$ используют алгоритм $\mathcal{A}1$ в качестве основной квантовой процедуры. При этом искомое слово w представляется в схеме с использованием m кубит для $\mathcal{A}1$. Применение $\mathcal{A}1$ в качестве процедуры использует уже $\log m$ кубит для анализа вхождения слова w в $string$. □

В качестве комментария к алгоритму $\mathcal{A}3$ отметим, что его «большая квантовость» по сравнению с алгоритмом $\mathcal{A}2$ технически заключается в применении, по сути, квантовой генерации случайных последовательностей. Известно, что квантовые генераторы случайных последовательностей (ГСЧ) в некотором смысле являются идеальными физическими ГСЧ.

3. Техника универсального хеширования для квантового поиска

Понятие универсального хеширования определено в работе [13] и достаточно подробно рассматривалось в целом ряде работ (см., например, [14, 15]). Мы определим универсальное семейство хеш-функций в соответствии с нашими целями. Семейство $\mathcal{F} = \{f_1, \ldots, f_d\}$ словарных функций $f : \{0,1\}^m \to \{0,1\}^l$ для $l < m$ называется универсальным семейством хеш-функций (для области определений функций f – множества $\{0.1\}^m$), если для некоторого $\epsilon \in [0,1]$ и для произвольной пары $v, w \in \{0.1\}^m$ выполняется

$$\frac{|F_{v,w}|}{|\mathcal{F}|} \leq \epsilon,$$

где $F_{w,v} = \{f \in \mathcal{F} : f(v) = f(w)\}$.

Для наших целей дополним определение универсального семейства хеш-функций следующим образом.

Универсальное семейство $\mathcal{F} = \{f_1, \ldots, f_d\}$ хеш-функций $f : \{0,1\}^m \to \{0,1\}^l$ для $n \leq 2^m$ и $\epsilon \in [0,1]$ будем называть *сильно (n,ϵ)-универсальным семейством хеш-функций*, если для каждого n-подмножества $Set = \{v_1, \ldots, v_n\}$ множества $\{0.1\}^m$ и произвольного слова $w \in \{0.1\}^m$ выполняется

$$\frac{|F_{Set,w}|}{|\mathcal{F}|} \leq \epsilon,$$

где $F_{Set,w} = \bigcup\limits_{v \in Set} F_{v,w}$.

Заметим, что сильно $(1,\epsilon)$-универсальное семейство хеш-функций является ϵ-универсальным семейством хеш-функций в стандартном понимании.

Примером сильно (n,ϵ)-универсального семейства хеш-функций для множеств $Set = T(string)$ и w является множество $\mathcal{F} = \{f_1, \ldots, f_d\}$, где функции f_j определяются j-м простым числом p_j так, что $f_j(w) = r(w)_{p_j}$ (слово w считается целым $w \in \{0, \ldots, 2^m - 1\}$, а $r(w)_{p_j}$ – это (представленный в виде слова) остаток r от деления w на простое p_j). В силу доказательства теоремы 3 имеем, что для константы $c \geq 3$ b и $d = cnm$ множество $\mathcal{F} = \{f_1, \ldots, f_d\}$ является сильно $(n, 1/c)$-универсальным семейством хеш-функций для каждого множеств $Set \subseteq \{0,1\}^m$ мощности n и каждого слова $w \in \{0.1\}^m$.

3.1. Алгоритм $\mathcal{A}2$ в терминах сильно универсального семейства хеш-функций. В терминах сильно (n,ϵ)-универсального семейства хеш-функций $\mathcal{F} = \{f_1, \ldots, f_d\}$ алгоритм $\mathcal{A}2$ описывается следующим образом.

Первый этап алгоритма – классический: для множества $T(string)$ $(T(string) = n)$ и слова w выбирается сильно $(n.\epsilon)$-универсальное семейство хеш-функций $\mathcal{F} = \{f_1, \ldots, f_d\}$. Равновероятностно выбирается функция f из множества $\mathcal{F}$.

Второй этап (*подготовка квантового состояния*): для функции $f \in \mathcal{F}$ и $string$. Через

$$T_f(string) = \{f(w_0) \ldots, f(w_{n-1})\}$$

обозначим последовательность значений функции f на элементах $string$. По $T_f(string)$ порождается квантовое состояние

$$|string, f\rangle = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} |k\rangle \otimes |f(w_k)\rangle \otimes |1\rangle.$$

Третий этап (квантовый) алгоритма: к состоянию $|string, f\rangle$ и искомому слову w применяется алгоритм $\mathcal{A}1$. В результате измерения первых $\log n$ кубит будет получен номер $j \in \{0, \ldots, n-1\}$, который выдается в качестве ответа и объявляется искомым номером слова w_j такого, что $w_j = w$.

Применение понятия сильно универсального семейства хеш-функций в конструкции алгоритма $\mathcal{A}2$ дает нам следующее обобщения теоремы 3.

Теорема 6. *Для текста string, его n-множества $T(string)$, слова w длины m пусть семейство $\mathcal{F}$ является сильно (n, ϵ)-универсальным семейством хеш-функций $\mathcal{F} = \{f_1, \ldots, f_d\}$, где $f_j : \{0,1\}^m \to \{0,1\}^l$. Пусть алгоритм $\mathcal{A}2$ реализован на основе $\mathcal{F}$. Тогда*

$$Er^{\mathcal{A}2}(string, w)) \leq \epsilon + \frac{1}{n},$$
$$Q^{\mathcal{A}2}(n, m) = O(l\sqrt{n}),$$
$$S^{\mathcal{A}2}(n, m) \leq O(\log n + l)$$

Аналогично модифицируется утверждение теоремы 4 для алгоритма $\mathcal{A}3$.

4. Техника квантового хеширования для поиска

Понятие квантового хеширования, его свойства исследовались в ряде работ нашей группы [16–18].

Здесь мы определяем квантовую функцию в соответствии с потребностями настоящей работы.

Для $\{0,1\}^m$ и множества $(\mathcal{H}^2)^{\otimes s}$ квантовых s кубитных состояний функцию

$$\psi : \{0,1\}^m \to (\mathcal{H}^2)^{\otimes s}$$

будем называть квантовой нерастягивающей (сжимающей) функцией, если $s \leq m$ $(s < m)$. Квантовая хеш-функция определялась нами с намерением рассматривать криптографические аспекты хеширования. Здесь нас интересуют возможности квантовых функций как «сжимающих отображений» со свойством « устойчивости к коллизиям». Эти два свойства лежат в основе применений хеширования. Понятие «квантовая устойчивая к коллизиям функция» определим следующим образом.

Для $0 \leq \epsilon \leq 1$ квантовую функцию ψ будем называть ϵ-устойчивой к коллизиям, если для каждой пары w, w' различных элементов из $\mathbb{X}$ скалярное произведение состояний $|\psi(w)\rangle, |\psi(w')\rangle$ удовлетворяет неравенству

$$|\langle\psi(w)|\psi(w')\rangle| \leq \epsilon.$$

Состояния $|\psi(w)\rangle$ и $|\psi(w')\rangle$ будем называть ϵ-ортогональными.

Требование ϵ-ортогональности – аналог устойчивости к коллизиям для классических хеш-функций [18].

Собственно в терминах данного раздела алгоритм $\mathcal{A}1$ применяет нерастягивающую 0-устойчивую квантовую хеш-функцию

$$\psi\{0,1\}^m \to (\mathcal{H}^2)^{\otimes m} \quad \text{вида} \quad \psi : w \mapsto |w\rangle.$$

Имеем $\langle\psi(w)\psi(w')\rangle = 0$. При этом отметим, что оракульное преобразование O_{f_w} алгоритма $\mathcal{A}1$ задается на основе хеш-функции ψ алгоритма.

Заключение

В работе представлены три квантовых алгоритма $\mathcal{A}1$, $\mathcal{A}2$ и $\mathcal{A}3$ поиска вхождения слова w в текст $string$. Алгоритм $\mathcal{A}1$ – это по сути алгоритм квантового поиска Гровера, соответствующий задаче. Здесь мы используем оригинальные условия применения алгоритма Гровера, а именно, считаем, что в тексте $string$ имеется (обязательно) единственное вхождение слова w. Основной (заявленный) результат экономии числа используемых кубит достигается в алгоритмах $\mathcal{A}2$ и $\mathcal{A}3$ с применением алгоритма $\mathcal{A}1$ в качестве основной квантовой процедуры. Показаны возможности применения хеширования – универсального хеширования и техники квантового хеширования. По сути, как это показано в разд. 3 и 4, выигрыш в числе используемых кубит в алгоритмах $\mathcal{A}2$ и $\mathcal{A}3$ достигается при применении квантового хеширования и использования свойств универсального хеширования.

Важно отметить, что в настоящей работе, как и в работе [9], алгоритмы применяются к квантовому состоянию (начальному состоянию), которое заранее готовится на основе анализируемого текста $string$. Разумеется, подготовка начального состояния требует предварительной работы, но эта работа не учитывается в алгоритмах. Отметим, что проблема подготовки начального состояния, как особой задачи, начинает обсуждаться в научном сообществе, занимающемся квантовым поиском информации. В частности, на проблему сложности подготовки начального состояния обратили внимание авторы работы [19].

Еще раз отметим, что в настоящей работе рассматривался случай, когда заранее точно известно, что искомая подстрока встречается в тексте ровно один раз. Задача может быть расширена, если количество вхождений больше 1 и заранее известно, а также если количество вхождений подстроки заранее неизвестно. Оценка временной и пространственной сложности алгоритма поиска Гровера в этих случаях описана в работе [11].

Благодарности. Исследование выполнено за счет гранта Российского научного фонда (проект № 19-19-00656).

Литература

1. *Китаев А., Шень А., Вялый М.* Классические и квантовые вычисления. – М.: МЦНМО, 1999. – 192 с.
2. *Grover L.K.* A fast quantum mechanical algorithm for database search // Proc. 28th Annu. ACM Symp. on Theory of Computing. – 1996. – P. 212–219. – doi: 10.1145/237814.237866.
3. *Gilliam A., Pistoia M., Gonciulea C.* Optimizing quantum search using a generalized version of Grover's algorithm. – 2020. – arXiv:2005.06468.
4. *Brassard G., Hoyer P., Mosca M., Tapp A.* Quantum amplitude amplification and estimation // Samuel J. Lomonaco Jr. (Eds.) AMS Contemporary Mathematics. – 2002. – V. 305. – P. 53–74.
5. *Reitzner D., Nagaj D., Buzek V.* Quantum walks. – 2012. – arXiv:1207.7283.
6. *Ablayev F., Ablayev M., Huang J.Zh., Khadiev K., Salikhova N., Wu D.* On quantum methods for machine learning problems part I: Quantum tools // Big Data Min. Anal. – 2019. – V. 3, No 1. – P. 41–55. – doi: 10.26599/BDMA.2019.9020016.
7. *Ablayev F., Ablayev M., Huang J.Zh., Khadiev K., Salikhova N., Wu D.* On quantum methods for machine learning problems part II: Quantum classification algorithms // Big Data Min. Anal. – 2019. – V. 3, No 1. – P. 56–67. – doi: 10.26599/BDMA.2019.9020018.

8. *Knuth D.E., Morris J.H. Jr., Pratt V.R.* Fast pattern matching in strings // SIAM J. Comput. – 1977. – V. 6, No 2. – P. 323–350. – doi: 10.1137/0206024.

9. *Ramesh H., Vinay V.* String matching in $\widetilde{O}(\sqrt{n}+\sqrt{m})$ quantum time // J. Discrete Algorithms. – 2003. – V. 1, No 1. – P. 103–110. – doi: 10.1016/S1570-8667(03)00010-8.

10. *Ambainis A.* Understanding quantum algorithms via query complexity // Proc. Int. Congress of Mathematicians (ICM 2018). – 2019. – P. 3265–3285. – doi: 10.1142/9789813272880_0181.

11. *Boyer M., Brassard G., Høyer P., Tapp A.* Tight bounds on quantum searching // Fortschr. Phys.: Prog. Phys.. – 1998. – V. 46, No 4–5. – P. 493–505. – doi: 10.1002/(SICI)1521-3978(199806)46:4/5<493::AID-PROP493>3.0.CO;2-P.

12. *Zhang K., Korepin V.E.* Examples on quantum search algorithm with optimized depth // Phys. Rev. A. – 2020. – V. 101, No 3. – Art. 032346, P. 1–12. – doi: 10.1103/PhysRevA.101.032346.

13. *Carter J.L., Wegman M.N.* Universal classes of hash functions // J. Comput. System Sci. – 1979. – V. 18, No 2. – P. 143–154. – doi: 10.1016/0022-0000(79)90044-8.

14. *Stinson D.R.* Universal hashing and authentication codes // Des. Codes Cryptogr. – 1994. – V. 4. – P. 369–380. – doi: 10.1007/BF01388651.

15. *Stinson D.R.* Combinatorial techniques for universal hashing // J. Comput. Syst. Sci. – 1994. – V. 48, No 2. – P. 337–346. – doi: 10.1016/S0022-0000(05)80007-8.

16. *Ablayev F.M., Vasiliev A.V.* Cryptographic quantum hashing // Laser Phys. Lett. – 2013. – V. 11, No 2. – Art. 025202, P. 1–4. – doi: 10.1088/1612-2011/11/2/025202.

17. *Ablayev F., Ablayev M.* On the concept of cryptographic quantum hashing // Laser Phys. Lett. – 2015. – V. 12, No 12. – Art. 125204, P. 1–5. – doi: 10.1088/1612-2011/12/12/125204.

18. *Аблаев Ф.М., Аблаев М.Ф., Васильев А.В.* Универсальное квантовое хеширование // Учен. зап. Казан. ун-та. Сер. Физ.-матем. науки. – 2014. – Т. 156, кн. 3. – С. 7–18.

19. *Macaluso A., Clissa L., Lodi St., Sartori C.* Quantum Ensemble for Classification. – 2020. – arXiv:2007.01028.

Поступила в редакцию
23.07.2020

Аблаев Марат Фаридович, младший научный сотрудник лаборатории нелинейной оптики; научный сотрудник лаборатории «Квантовые методы обработки информации»
Казанский физико-технический институт им. Е.К. Завойского ФИЦ Казанский научный центр РАН
ул. Сибирский тракт, д. 10/7, г. Казань, 420029, Россия
Казанский (Приволжский) федеральный университет
ул. Кремлевская, д. 18, г. Казань, 420008, Россия
E-mail: *mablayev@gmail.com*

Салихова Наиля Маратовна, младший научный сотрудник лаборатории «Квантовые методы обработки информации»
Казанский (Приволжский) федеральный университет
ул. Кремлевская, д. 18, г. Казань, 420008, Россия
E-mail: *nailyasalikhova66@gmail.com*

ISSN 2541-7746 (Print)
ISSN 2500-2198 (Online)

**UCHENYE ZAPISKI KAZANSKOGO UNIVERSITETA.
SERIYA FIZIKO-MATEMATICHESKIE NAUKI
(Proceedings of Kazan University. Physics and Mathematics Series)**

2020, vol. 162, no. 3, pp. 241–258

doi: 10.26907/2541-7746.2020.3.241-258

Quantum Search for a Given Substring in the Text Using a Hashing Technique

M.F. Ablayev[a,b*], *N.M. Salikhova*[b**]

[a] *Zavoisky Physical-Technical Institute, FRC Kazan Scientific Center, Russian Academy of Sciences, Kazan, 420029 Russia*
[b] *Kazan Federal University, Kazan, 420008 Russia*
E-mail: **mablayev@gmail.com*, ***nailyasalikhova66@gmail.com*

Received July 23, 2020

Abstract

The problem of searching for a given substring in the text was considered. It is known that classical algorithms solve this problem in a linear time depending on the length of the text and the specified template. Quantum algorithms speed up the search by "square root times".

In this paper, we proposed a quantum algorithm that solves the search problem a) with a high probability of getting the correct result and b) with the same acceleration (by "square root times") as compared with the classical one, but it c) requires much less memory (based on the number of qubits used) than the previously known quantum algorithms.

Keywords: quantum algorithms, string search, quantum search

Acknowledgments. The study was supported by the Russian Science Foundation (project no. 19-19-00656).

Figure Captions

Fig. 1. General quantum circuit of the oracle in the algorithm of searching for a substring.

Fig. 2. Quantum circuit of the oracle in the algorithm of searching for the substring $01\ldots0$.

Fig. 3. Circuit of the operator $\mathcal{D}$.

Fig. 4. Quantum circuit of the iteration of Grover's algorithm.

Fig. 5. Quantum circuit of the searching algorithm.

References

1. Kitaev A., Shen' A., Vyalyi M. *Klassicheskie i kvantovye vychisleniya* [Classical and Quantum Computation]. Moscow, MTsNMO, 1999. 192 p. (In Russian)
2. Grover L.K. A fast quantum mechanical algorithm for database search. *Proc. 28th Annu. ACM Symp. on Theory of Computing*, 1996, pp. 212–219. doi: 10.1145/237814.237866.
3. Gilliam A., Pistoia M., Gonciulea C. Optimizing quantum search using a generalized version of Grover's algorithm. 2020, arXiv:2005.06468.

4. Brassard G., Hoyer P., Mosca M., Tapp A. Quantum amplitude amplification and estimation. In: Samuel J. Lomonaco Jr. (Eds.) *AMS Contemporary Mathematics*. 2002, vol. 305, pp. 53–74.

5. Reitzner D., Nagaj D., Buzek V. Quantum walks. 2012, arXiv:1207.7283.

6. Ablayev F., Ablayev M., Huang J.Zh., Khadiev K., Salikhova N., Wu D. On quantum methods for machine learning problems part I: Quantum tools. *Big Data Min. Anal.*, 2019, vol. 3, no. 1, pp. 41–55. doi: 10.26599/BDMA.2019.9020016.

7. Ablayev F., Ablayev M., Huang J.Zh., Khadiev K., Salikhova N., Wu D. On quantum methods for machine learning problems part II: Quantum classification algorithms. *Big Data Min. Anal.*, 2019, vol. 3, no. 1, pp. 56–67. doi: 10.26599/BDMA.2019.9020018.

8. Knuth D.E., Morris J.H. Jr., Pratt V.R. Fast pattern matching in strings. *SIAM J. Comput.*, 1977, vol. 6, no. 2, pp. 323–350. doi: 10.1137/0206024.

9. Ramesh H., Vinay V. String matching in $\widetilde{O}(\sqrt{n}+\sqrt{m})$ quantum time. *J. Discrete Algorithms*, 2003, vol. 1, no. 1, pp. 103–110. doi: 10.1016/S1570-8667(03)00010-8.

10. Ambainis A. Understanding quantum algorithms via query complexity. *Proc. Int. Congr. of Mathematicians (ICM 2018)*, 2019, pp. 3265–3285. doi: 10.1142/9789813272880_0181.

11. Boyer M., Brassard G., Høyer P., Tapp A. Tight bounds on quantum searching. *Fortschr. Phys.: Prog. Phys.*, 1998, vol. 46, nos. 4–5, pp. 493–505. doi: 10.1002/(SICI)1521-3978(199806)46:4/5¡493::AID-PROP493¿3.0.CO;2-P.

12. Zhang K., Korepin V.E. Examples on quantum search algorithm with optimized depth. *Phys. Rev. A.*, 2020, vol. 101, no. 3, art. 032346, pp. 1–12. doi: 10.1103/PhysRevA.101.032346.

13. Carter J.L., Wegman M.N. Universal classes of hash functions. *J. Comput. Syst. Sci.*, 1979, vol. 18, no. 2, pp. 143–154. doi: 10.1016/0022-0000(79)90044-8.

14. Stinson D.R. Universal hashing and authentication codes. *Des. Codes Cryptogr.*, 1994, vol. 4, pp. 369–380. doi: 10.1007/BF01388651.

15. Stinson D.R. Combinatorial techniques for universal hashing. *J. Comput. Syst. Sci.*, 1994, vol. 48, no. 2, pp. 337–346. doi: 10.1016/S0022-0000(05)80007-8.

16. Ablayev F.M., Vasiliev A.V. Cryptographic quantum hashing. *Laser Phys. Lett.*, 2013, vol. 11, no. 2, art. 025202, pp. 1–4. doi: 10.1088/1612-2011/11/2/025202.

17. Ablayev F., Ablayev M. On the concept of cryptographic quantum hashing. *Laser Phys. Lett.*, 2015, vol. 12, no. 12, art. 125204, pp. 1–5. doi: 10.1088/1612-2011/12/12/125204.

18. Ablayev F.M., Ablayev M.F., Vasilev A.V. Universal quantum hashing. *Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki*, 2014, vol. 156, no. 3, pp. 7–18. (In Russian)

19. Macaluso A., Clissa L., Lodi St., Sartori C. *Quantum Ensemble for Classification*. 2020, arXiv:2007.01028.

⟨ ***Для цитирования*:** *Аблаев М.Ф., Салихова Н.М.* Квантовый поиск заданной подстроки в тексте на основе техники хеширования // Учен. зап. Казан. ун-та. Сер. Физ.-матем. науки. – 2020. – Т. 162, кн. 3. – С. 241–258. – doi: 10.26907/2541-7746.2020.3.241-258. ⟩

⟨ ***For citation*:** Ablayev M.F., Salikhova N.M. Quantum search for a given substring in the text using a hashing technique. *Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki*, 2020, vol. 162, no. 3, pp. 241–258. doi: 10.26907/2541-7746.2020.3.241-258. (In Russian) ⟩