УДК 004.415.2, 004.588

МОДЕЛИРОВАНИЕ КВАНТОВОГО ПОИСКОВОГО АЛГОРИТМА ШОРА НА КЛАССИЧЕСКОМ КОМПЬЮТЕРЕ[1]

Петров Сергей Павлович[1], Ульянов Сергей Викторович[2]

[1] Студент;
ГОУ ВПО «Международный Университет природы, общества и человека «Дубна»,
Институт системного анализа и управления;
141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: bloodthirsty_89@mail.ru.

[2] Доктор физико-математических наук, профессор;
ГОУ ВПО «Международный Университет природы, общества и человека «Дубна»,
Институт системного анализа и управления;
141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: ulyanovsv@mail.ru.

Показана возможность выполнения квантовых вычислений на классическом компьютере. Описана структура и порядок действий при моделировании квантовых алгоритмов на примере квантового поискового алгоритма Шора.

Ключевые слова: квантовые вычисления, алгоритм Шора, факторизация.

SHOR'S QUANTUM SEARCHING ALGRITHM SIMULATION ON THE CLASSICAL COMPUTER

Petrov Sergey[1], Ulyanov Sergey[2]

[1] Student;
Dubna International University of Nature, Society, and Man,
Institute of system analysis and management;
141980, Dubna, Moscow reg., Universitetskaya str., 19;
e-mail: bloodthirsty_89@mail.ru.

[2] Doctor of Science Physics and Mathematics, professor;
Dubna International University of Nature, Society, and Man,
Institute of system analysis and management;
141980, Dubna, Moscow reg., Universitetskaya str., 19;
e-mail: ulyanovsv@mail.ru.

The ability of quantum computation performing on the classical computer is presented. The structure and the operations' order of quantum algorithms simulation in accordance with the example of Shor's quantum searching algorithm is described.

Keywords: quantum computing, Shor's algorithm, factorisation.

Введение

Начиная с семидесятых годов прошлого столетия, интенсивно осуществляется поиск эффективных алгоритмов разложения целого числа на множители. Однако даже самые быстрые из известных алгоритмов являются экспоненциальными по сложности в зависимости от размерности входа – количества разрядов целого числа N. В результате среди специалистов утвердилось мнение об отсутствии

[1] Работа отмечена Первой степенью секцией «Информационные технологии» 17-й научно-практической конференции студентов, аспирантов и молодых специалистов Университета «Дубна», апрель 2010.

эффективного способа разложения на множители, а сложность этой проблемы принята за основу надежности многих криптографических систем, например *RSA* (система шифрования с открытым ключом).

В 1994 году Питер Шор разработал вероятностный алгоритм разложения на множители *n*-разрядных чисел за полиномиальное время на квантовом компьютере. Это стало открытием не только из-за появления потенциальной угрозы информационной безопасности, но еще и потому, что ее решение показало вычислительное превосходство квантовых систем над классическими системами (хотя бы на некотором множестве специализированных задач). Впоследствии, было открыто множество квантовых алгоритмов и доказана их эффективность в сравнение с традиционными вычислениями. Но до некоторых пор считалось, что такая эффективность может быть полезной только при наличии квантового компьютера.

В настоящее время существует тенденция к моделированию квантовых вычислений на классическом компьютере и разработано большое количество различных подходов к моделированию. Не так давно был разработан относительно эффективный метод программной реализации квантовых алгоритмов на классическом компьютере [1, 2].

В данной статье описывается универсальный метод моделирования квантовых алгоритмов на классическом компьютере, а также приводится описание работы алгоритма Шора и описывается порядок моделирования его работы предложенным методом на компьютере с архитектурой фон Неймана.

Описание квантового алгоритма Шора

Алгоритм Шора причисляется к классу квантовых поисковых алгоритмов за его способность находить множители целого числа в конечном пространстве поиска, но он отличается от других алгоритмов тем, что задача квантового поиска вытекает из факторизации косвенным способом. Вычисление множителей происходит путем выявления периода специальной функции (1), использующей само число, требуемое к разложению, в качестве параметра[2]:

$$f_{N,a} : \{0,1\}^n \to \{0,1\}^n : f_{N,a}(x) = a^x \bmod N, \tag{1}$$

где N – целое число, множители которого требуется найти, a — случайное целое положительное число меньшее N: $НОД(a,N) \equiv 1,\ x = 0,1\ldots,2^n - 1,$ где $n = \lceil \log_2 N \rceil + 1.$ N и a являются параметрами для функции (1). Заметим, что требование к a быть взаимно простым с N вызвано тем, что в противном случае нам уже известны делители числа N. Если мы найдем период r функции (1), то в случае, когда r нечетно, N – простое число, а в случае, когда r четно, выполняется следующее:

$$\text{т.к. } a^0 \equiv 1(mod\ N) \Rightarrow a^r \equiv 1(mod\ N) \Rightarrow$$

$$\left(a^{2\frac{r}{2}} - 1\right) \equiv 0(mod\ N) \Rightarrow \left(a^{\frac{r}{2}} - 1\right)\left(a^{\frac{r}{2}} + 1\right) \equiv 0(mod\ N) \Leftrightarrow \left(a^{\frac{r}{2}} - 1\right)\left(a^{\frac{r}{2}} + 1\right) \equiv hN,$$

где h – целое положительное число. Из этих преобразований следует, что $\left(a^{\frac{r}{2}} - 1\right)$ или $\left(a^{\frac{r}{2}} + 1\right)$ должен иметь общий нетривиальный делитель с N, который можно вычислить, использую например алгоритм Евклида, обладающий полиномиальной по времени сложностью[3].

Таким образом, для разложения нам необходимо найти период функции (1).

Квантовый алгоритм Шора использует два квантовых регистра X и Y, первоначально находящихся в нулевом состоянии $|0\rangle$. Размер первого регистра – n, достаточный для размещения аргументов функции, размер второго (вспомогательного) регистра выбирается так, чтобы можно было записать любое значение самой функции. Часто значение длины регистра Y также принимается равным n, сле-

[2] В 1976 году Миллер доказал, что задача о факторизации, использующей рандомизацию, может быть сведена к задаче о нахождения периода функции (порядка функции).

[3] Заметим, что эти вычисления выполняются классическим образом.

довательно, число состояний каждого регистра 2^n. Рассмотрим итерации в квантовом алгоритме Шора для нахождения порядка *r*.

Шаг 1. Первый шаг алгоритма – создание равновероятной суперпозиции всех входов, всех значений $x \in \left[0, 2^n\right)$. Данная операция выполняется путем применения оператора Уолша-Адамара. Значение второго регистра хранит значения функции (1). Получим следующее квантовое состояние:

$$\psi = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} | x \otimes | a^x \bmod N > . \tag{2}$$

Шаг 2. Является подготовительным для квантового преобразования Фурье, проводимого в шаге три. Чтобы использовать это преобразование для получения периода $f_{N,a}$, необходимо подготовить состояние, амплитуда функции которого имеет тот же период, что и $f_{N,a}$. Для этого необходимо измерить состояние *второго* регистра. Это измерение проектирует пространство состояний системы из *n* кубитов на подпространство данной измеренной величины. Другими словами, измеряя состояние регистра *Y*, мы получаем периодическое состояние в первом регистре. После измерения получим $C\sum_x g(x) | x, u >$, где [4]:

$$g(x) = \begin{cases} 1, \ f_{N,a} = u \\ 0, \text{ в остальный случаях} \end{cases} \tag{3}$$

Шаг 3. Квантовое преобразование Фурье (*QFT)* определяется как

$$QFT : x \to \frac{1}{\sqrt{2^n}} \sum_{c=0}^{2^n-1} \exp(2\pi i c x / 2^n) | c > \tag{4}$$

Второй регистр больше не используется, опустим его запись. Применяя (4) к (3), при периоде функции (1) *r* равном степени два, получим $\sum_j c_j | j \frac{2^n}{r}$, где амплитуда равна нулю во всех точках кроме точек кратных $\frac{2^n}{r}$. Когда же период *r* не делится на 2^n, преобразования выполняются приближенно. Причем наибольшая амплитуда сосредоточена вблизи целых значений кратных $\frac{2^n}{r}$.

Шаг 4. Измеряем конечное состояние регистра *X* и получаем результат *v*. Когда квантовое преобразование выполняется точно (когда *r* является степенью два), имеем:

$$v = j \frac{2^n}{r} . \tag{5}$$

При этом сокращение дроби $\frac{v}{2^n} = \frac{j}{r}$ даст искомый *r* как знаменатель[5]. В случае же приближенных преобразований Фурье, используя механизм разложения в бесконечную дробь, найдем знаменатель $\tilde{q}$, который будет хорошим приближением периода *r*.

Моделирование алгоритма Шора на классическом компьютере

[4] Данный шаг можно исключить, т.к. квантовое преобразование Фурье, применяемое к суперпозиции нескольких периодических функций, приводит к суперпозиции преобразований Фурье этих функций. Каждое из этих преобразований связано с соответствующим значением *u*; следовательно, они не интерферируют друг с другом [3 - 6].

[5] В большинстве случаев *r* и *j* будут взаимно просты. Если же они будут иметь общий множитель, знаменатель будет являться делителем периода, а не самим периодом.

На рис. 1 изображена общая схема моделирования квантовых алгоритмов на классическом компьютере [1].

Рассмотрим моделирование квантового алгоритма Шора по данной схеме.

Итак, чтобы определить множители целого числа *N*, необходимо найти период *r* функции (1). Задача поиска множителей переходит в задачу поиска периода *r* периодической функции $f:\{0,1\}^n \to \{0,1\}^n$.

Кодирование. На рис. 2 представлена схема, по которой работает блок кодирования.

Функция f преобразуется в унитарный оператор U_F последовательным выполнением следующих действий:

Шаг 1. Таблица отображения функции $f:\{0,1\}^n \to \{0,1\}^n$ преобразуется в таблицу отображения унитарной функции $F:\{0,1\}^{n+n} \to \{0,1\}^{n+n}$, такой что:

$$F(x_0,...,x_{n-1},y_0,...,y_{n-1}) = (x_0,...,x_{n-1},f(x_0,...,x_{n-1}) \oplus (y_0,...,y_{n-1})).$$

Необходимость данного преобразования связана с требованием унитарности оператора U_F. Оператор U_F является обратимым, следовательно, он не может отображать два разных входа в одинаковые выходные значения. Так как данный оператор является матричным представлением функции F, функция F должна быть обратимой функцией. Если мы опустим создание функции F и напрямую создадим матричный оператор U_f, то он не будет являться унитарным, так как функция f необязательно обратима.

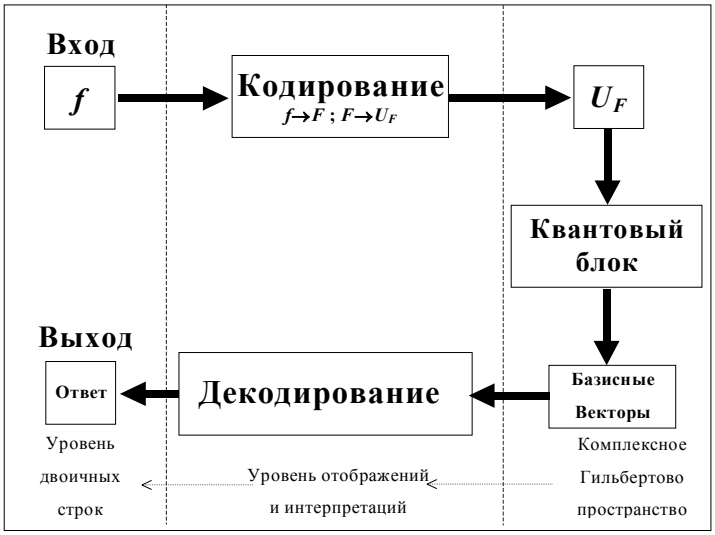


Рис. 1. Схематическая диаграмма квантовых алгоритмов

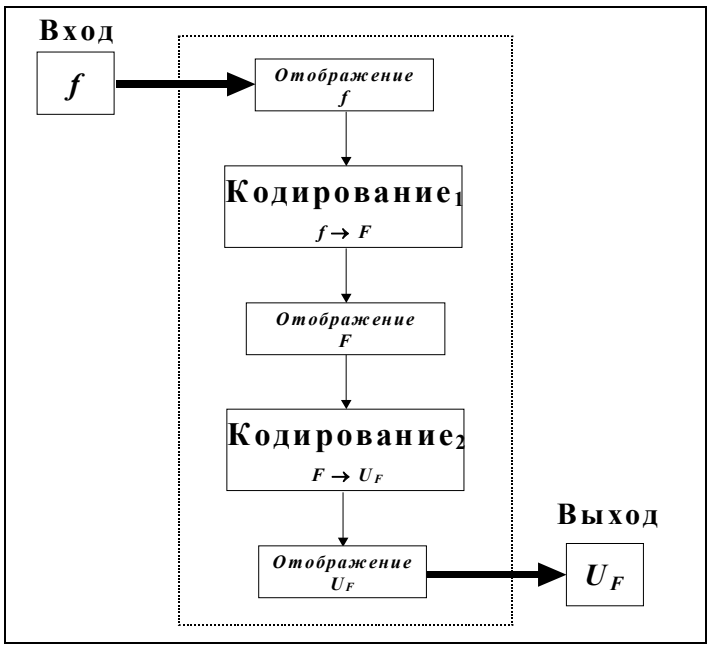


Рис. 2. Диаграмма блока кодирования

Таким образом, обратимость выполняется за счет увеличения числа бит и рассмотрением функции F вместо функции f. В любом случае, функция f может быть всегда получена из F вычислением последних n значений при $(y_0,...,y_{n-1}) = (0,...,0)$.

<u>*Шаг* 2</u>. Отображение функции F преобразуется в отображение U_F, согласно следующим ограничениям: $\forall s \in \{0,1\}^{2n} : U_F[\tau(s)] = \tau[F(s)]$. Таблица кодирования $\tau : \{0,1\}^{2n} \to C^{2^{2n}}$, где $C^{2^{2n}}$ есть результирующее Гильбертово пространство, определяется как:

$$\tau(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \quad \tau(1) = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

$$\tau(x_0,...,x_{2n-1}) = \tau(x_0) \otimes ... \otimes \tau(x_{2n-1}) = |x_0 ... x_{2n-1}\rangle.$$

Кодирование τ преобразует битовые значения в комплексные векторы размерности 2, принадлежащие каноническому базису комплексного Гильбертова пространства C^2.

<u>*Шаг* 3</u>. Отображение U_F преобразуется в матричный оператор U_F, следуя следующему правилу: $[U_F]_{ij} = 1 \Leftrightarrow U_F|i\rangle = |j\rangle$. Данное правило легко понять, если рассмотреть $|i\rangle$ и $|j\rangle$ как векторы-столбцы. Распределяя эти векторы-столбцы по каноническому базису, U_F определяет перестановку рядов матрицы идентичности. В общем виде, ряд $|j\rangle$ отображается в ряд $|i\rangle$.

Пример 1

Рассмотрим пример: пусть для функции (1) имеем следующие параметры *N*=4, a=3 $\Rightarrow$ n = 2. Тогда $f(x) = 3^x \, mod \, 4$, имеет следующую таблицу отображения:

(x_0, x_1)	$f(x_0, x_1)$
00	01
01	11
10	01
11	11

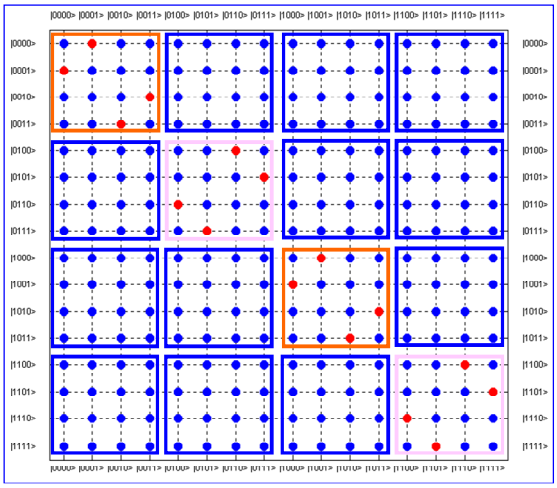
Из этой таблицы видно, что период функции $r = 2$. Теперь, согласно шагу 1, закодируем f в инъективную функцию F:

(x0,.., xn-1, y0,.., yn-1)	F(x0,.., xn-1, y0,.., yn-1)	(x0,.., xn-1, y0,.., yn-1)	F(x0,.., xn-1, y0,.., yn-1)
0000	0001	0010	0011
0100	0111	0110	0101
1000	1001	1010	1011
1100	1111	1110	1101
0001	0000	0011	0010
0101	0110	0111	0100
1001	1000	1011	1010
1101	1110	1111	1100

Далее закодируем таблицу отображения F в таблицу отображения U_F так, как это описано в шаге 2:

|x0.. xn-1 y0.. yn-1>	UF|x0.. xn-1 y0.. yn-1>	|x0.. xn-1 y0.. yn-1>	UF|x0.. xn-1 y0.. yn-1>
|0000>	|0001>	|0010>	|0011>
|0100>	|0111>	|0110>	|0101>
|1000>	|1001>	|1010>	|1011>
|1100>	|1111>	|1110>	|1101>
|0001>	|0000>	|0011>	|0010>
|0101>	|0110>	|0111>	|0100>
|1001>	|1000>	|1011>	|1010>
|1101>	|1110>	|1111>	|1100>

Получим матрицу U_F так, как это описано в шаге 3 (синие точки – нули, красные – единицы):



Или, идентично:

U_F	|00>	|01>	|10>	|11>
|00>	I⊗C	0	0	0
|01>	0	C⊗I	0	0
|10>	0	0	I⊗C	0
|11>	0	0	0	C⊗I

Таким образом, мы получили матрицу так называемого оператора запутывания.

В общем случае, она выглядит так:

U_F	|0..0>	|0..1>	…	|1..1>
|0..0>	$M_{0..0}$	0	…	0
|0..1>	0	$M_{0..1}$	…	0
...	...	…	…	...
|1..1>	0	0	0	$M_{1..1}$

где $M_i = P_1 \otimes ... \otimes P_n$, $P_k \in \{I, C\}$, k=1,..,n и $M_i = M_j \Leftrightarrow (j = i \vee j = (i+r) \bmod N)$.

Квантовый блок

Основой квантового блока является квантовая ячейка (см. рис. 3), зависящая от свойств матрицы оператора U_F. Матричный оператор U_F, являющийся выходом блока кодирования, в данной структуре является входом квантового блока (рис. 1).

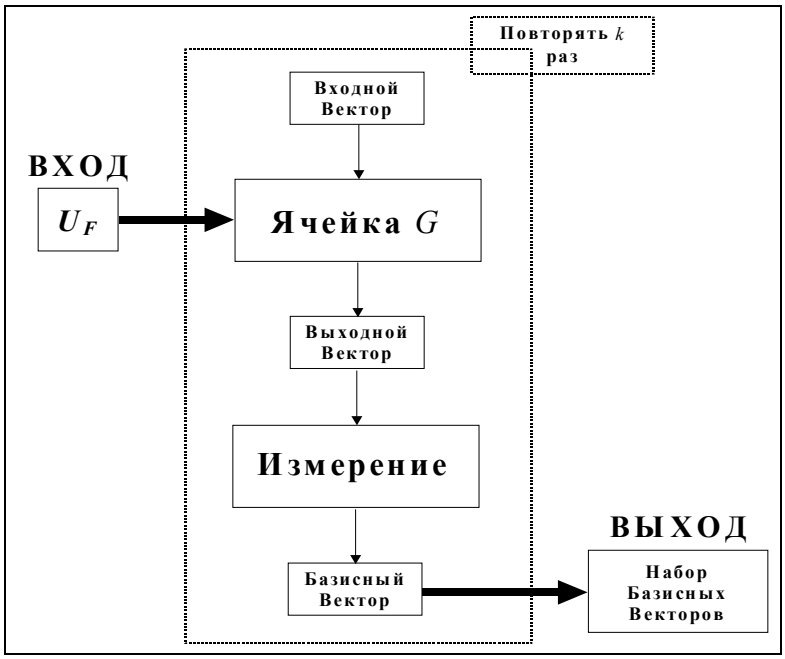


Рис. 3. Структура квантового блока (изображенного на рис. 1)

На первом этапе оператор U_F включается в более сложный оператор – квантовую ячейку G. Унитарная матрица G применяется k раз к начальному каноническому вектору $|i\rangle$ размерности 2^{2n}. Финальная комплексная линейная комбинация базисных векторов измеряется, и результатом имеем один базисный вектор $|x_i\rangle$. Все измеренные базисные векторы $\{x_1,..,x_k\}$ собираются вместе. Полученный набор является выходом квантового блока.

Ниже рассмотрим подробнее структуру квантовой ячейки и способ ее реализации на классическом компьютере.

Структура и принцип действия квантовой ячейки могут быть наглядно представлены квантовыми схемами на рис. 4. Квантовая ячейка представляет собой оператор-произведение трех матриц размерности $2^{2n} \times 2^{2n}$ каждая. Эти матрицы – ${}^n\mathrm{H} \otimes {}^n\mathrm{I}$, $\mathrm{U_F}$, $\mathrm{QFT}_n \otimes {}^n\mathrm{I}$ – матрицы операторов суперпозиции, запутывания и интерференции, соответственно. Перемножаются они в порядке, обратном порядку применения операторов.

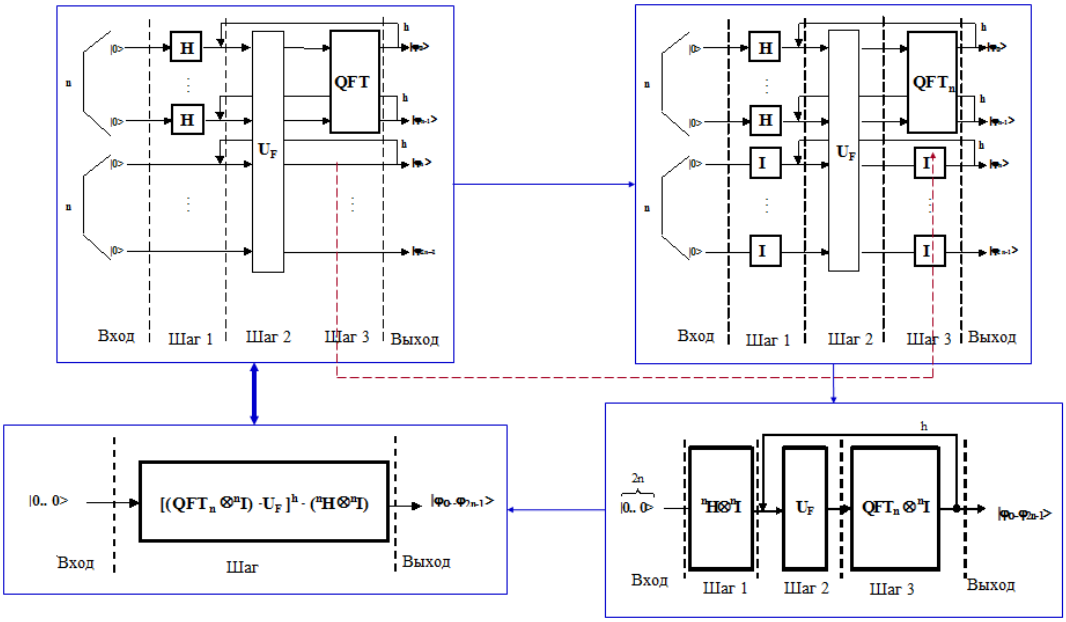


Рис. 4. Структура и принцип действия квантовой ячейки

Ниже рассмотрим подробнее, как получаются матрицы этих операторов.

Оператор суперпозиции

Рассматриваемый оператор схематично может быть представлен на рис. 5 и является тензорным произведением *n* операторов Адамара *H* и *n* операторов тождественного преобразования *I*.

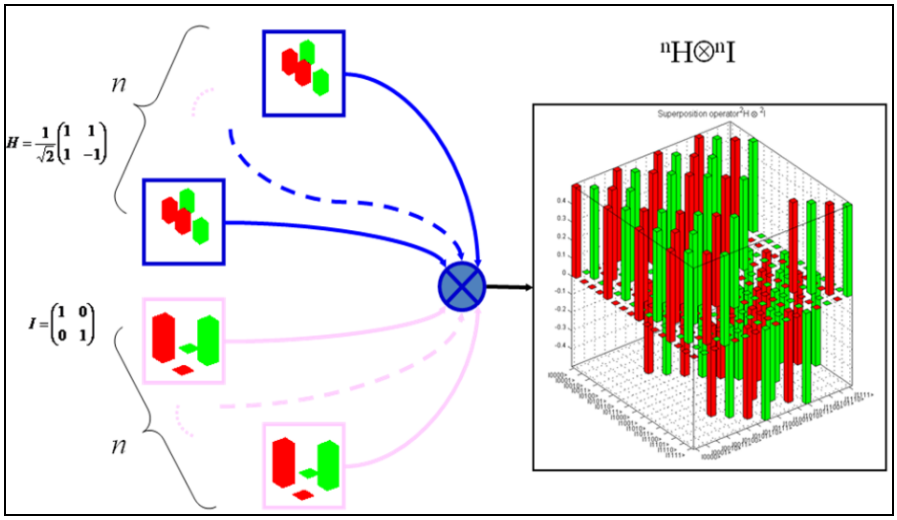


Рис. 5. Оператор суперпозиции

Полученный таким образом оператор действует на первый регистр (на первые *n* кубитов), создавая их суперпозицию, и действует тождественно на второй регистр (последние *n* кубитов), оставляя его без изменений.

Ниже показан общий вид матрицы оператора суперпозиции.

${}^nH \otimes {}^nI$	|0..0>	|0..1>	…	|j>	…	|1..1>
|0..0>	${}^nI/2^{n/2}$	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2}$
|0..1>	${}^nI/2^{n/2}$	$-{}^nI/2^{n/2}$	…	$(-1)^{(0..1)\cdot j}$ $({}^nI/2^{n/2})$	…	$-{}^nI/2^{n/2}$
…	…	…	…	…	…	…
|i>	${}^nI/2^{n/2}$	$(-1)^{i\cdot(0..1)}$ $({}^nI/2^{n/2})$	…	$(-1)^{i\cdot j}$ $({}^nI/2^{n/2})$	…	$(-1)^{i\cdot(1..1)}$ $({}^nI/2^{n/2})$
…	…	…	…	…	…	…
|1..1>	${}^nI/2^{n/2}$	$-{}^nI/2^{n/2}$	…	$(-1)^{(1..1)\cdot j}$ $({}^nI/2^{n/2})$	…	$(-1)^{(1..1)\cdot(1..1)}$ $({}^nI/2^{n/2})$

Оператором запутывания

Оператор запутывания является оператором U_F, полученный в результате работы блока кодирования.

В качестве *оператора интерференции* алгоритм Шора использует оператор $\mathrm{QFT}_n \otimes {}^n\mathrm{I}$, где QFT_n – квантовое преобразование Фурье первых n кубитов, ${}^n\mathrm{I}$ – тождественное преобразование последних n кубитов.

На рис. 6 схематично изображен этот оператор интерференции.

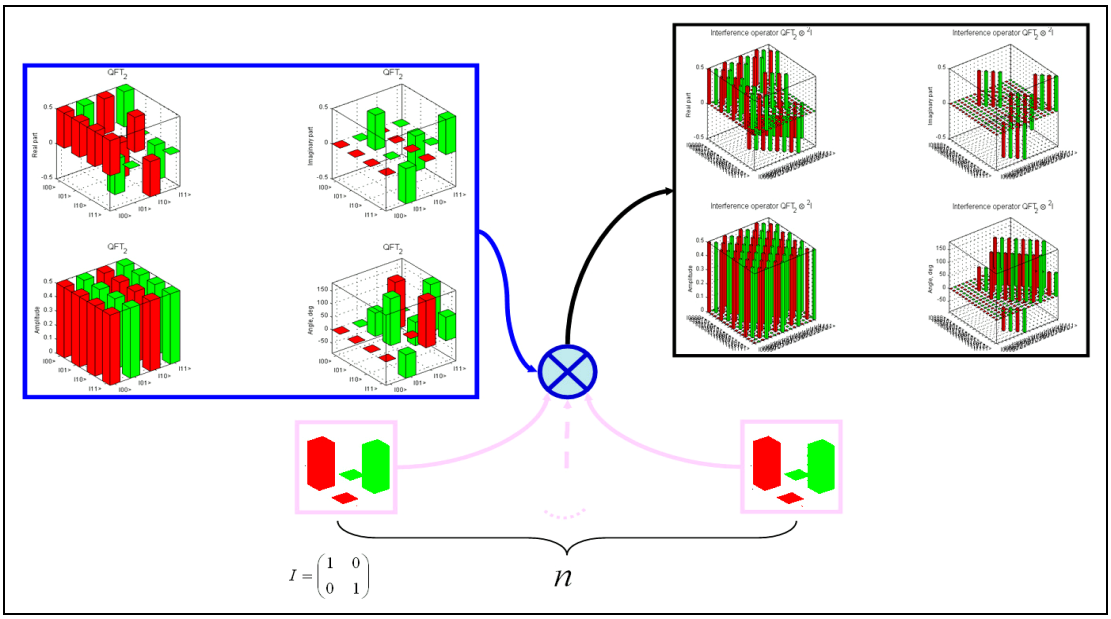


Рис. 6. Схема оператора интерференции

Оператор QFT – унитарный оператор, действующий на комплексных векторах Гильбертова пространства. Он преобразует входной вектор в суперпозицию базисных векторов с одинаковой амплитудой, но со смещенной фазой. Элементы оператора квантового преобразования Фурье могут быть вычислены как

$$\left[QFT_n\right]_{i,j} = \frac{1}{2^{n/2}} e^{2\pi J \frac{(i-1)(j-1)}{2^n}}, \text{ где } J \text{ - мнимая единица} \tag{6}$$

В соответствии с изложенным выше, общее математическое представление оператора интерференции представлено ниже.

$QFT_n \otimes {}^nI$	|0..0>	…	|j>	…	|1..1>
|0..0>	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2}$
…	…	…	…	…	…
|i>	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2} e^{J[i]_{10}\cdot[j]_{10}2\pi/2^n}$	…	${}^nI/2^{n/2}\ e^{J[i]_{10}\cdot(2^n-1)\,2\pi/2^n}$
…	…	…	…	…	…
|1..1>	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2} e^{J(2^n-1)\cdot[j]_{10}2\pi/2^n}$	…	${}^nI/2^{n/2}\ e^{J(2^n-1)^2\,2\pi/2^n}$

Сбор квантовой ячейки осуществляется, когда все операторы определены. Порядок применения этих операторов следующий: сначала применяется оператор суперпозиции (единожды), затем h раз последовательно применяются операторы запутывания и интерференции. Число h зависит от числа запусков квантового блока, т.е. от количества базисных векторов, которые должны быть получены в результате работы алгоритма:

$$G = \underbrace{\overbrace{QFT_n \otimes^n I}^{\text{Интерференция}} \cdot \underbrace{U_F}_{\text{Запутывание}} \cdot \cdots \cdot \overbrace{QFT_n \otimes^n}^{\text{Интерференция}} I \cdot \underbrace{U_F}_{\text{Запутывание}}}_{h \text{ раз}} \cdot \overbrace{{}^nH \otimes {}^nI}^{\text{Суперпозиция}} \quad (7)$$

Квантовая ячейка сама по себе представляет оператор, обобщенное математическое представление которого есть матрица вида:

G	|0..0>	…
|0..0>	$1/2^n\Sigma_{k\in\{0,1\}^n} e^{J\pi\cdot 0\cdot[k]_{10}/2^{n-1}} M_k$	…
…	…	…
|i>	$1/2^n\Sigma_{k\in\{0,1\}^n} e^{J\pi\cdot[i]_{10}\cdot[k]_{10}/2^{n-1}} M_k$	…
…	…	…
|1..1>	$1/2^n\Sigma_{k\in\{0,1\}^n} e^{J\pi\cdot(2^n-1)\cdot[k]_{10}/2^{n-1}} M_k$	…

(8)

Пример 2

Рассмотрим те же параметры функции (1), что и в первом примере. Последовательно выполняя шаги алгоритма, описанные выше, мы получим квантовую ячейку G такового вида:

G	|00>	|01>	|10>	|11>
|00>	(I⊗C+C⊗I)/2	(I⊗C-C⊗I)/2	0	0
|01>	0	0	(I⊗C+JC⊗I)/2	(I⊗C-JC⊗I)/2
|10>	(I⊗C-C⊗I)/2	(I⊗C+C⊗I)/2	0	0
|11>	0	0	(I⊗C-JC⊗I)/2	(I⊗C+JC⊗I)/2

Применяя данный оператор к базисному вектору, $|0000\rangle$, и производя измерения, мы можем получить состояния первого регистра: 00 — с вероятностью 0,5 и 10 с вероятностью 0,5 (в двоичном представлении). Рассмотрим расстояние $d = [|10 - 00|]_{10} = [10]_{10} = 2$, где $[s]_{10}$ – десятичное представление бинарной строки s. Заметим, что $N/r = 4/2 = 2$, т.е. $d = N/r$. Отсюда r может быть вычислено как $r = N/d$ [1].

Для общего случая имеем оператор G, представленной матрицей (8). Применяя оператор G к вектору $|0..0\rangle$, математически можно показать, что, измеряя результат и представляя далее его в двоичном виде, мы можем получить только такие бинарные строки i, что $im = l$, где m – некоторое целое число, а $l = \frac{2^n}{r}$. Тогда, $l \equiv 0(mod\, i)$ (9) [1].

Декодирование

Блок декодирования (см. рис. 1) имеет функцию интерпретации базисных векторов полученных после выполнения квантового блока. Декодирование базисных векторов заключается в их преобразо-

вании в бинарные строки с последующим использованием этих строк в качестве коэффициентов некоторого уравнения, либо в случае прямого кодирования решения задачи, для извлечения ответа. Задача декодирования в алгоритме Шора (как и в других алгоритмах) решается классическим алгоритмом: применяя квантовую ячейку к базисному вектору, мы получаем *h* базисных векторов, таких, что выполняется (9). Собирая эти векторы в систему уравнений, где компоненты векторов *i* служат коэффициентами уравнений, и решая эту систему, получим значение *l*. Так как $l = \frac{2^n}{r}$, мы можем вычислить искомый период: $r = \frac{2^n}{l}$.

Количество векторов, необходимое для того, чтобы получить *r*, зависит техники, с помощью которой решается система и, в общем случае, это значение возрастает как полином от *n*.

Нетрудно заметить, что количество кубитов, требуемых для решения задачи квантовым алгоритмом (размер регистров), играет критическую роль при вычислениях на классическом компьютере. Так, добавление только одного кубита подразумевает, что размерности матриц удваиваются по отношению к предыдущей конфигурации, а количество элементов и произведений увеличивается экспоненциально. Для того, чтобы избежать этой проблемы используется векторный подход и ряд алгоритмических упрощений, связанных со структурой операторов квантовой ячейки [7]. Рамки этой работы не позволяют привести описание данного подхода.

Заключение

Метод моделирования, приведенный в данной работе, дает возможность реализовать квантовый алгоритма Шора на классическом компьютере и выполнять факторизацию целых чисел. Очевидно, что метод, использующий прямые матричные произведения квантовых операторов, не является эффективным, он обладает такими полезными для нас качествами как универсальность (большинство квантовых алгоритмов может быть описано в терминах квантовой ячейки), наглядность, простота реализации.

Основная проблема программного моделирования квантовых алгоритмов состоит в невозможности эффективной реализации модели квантового параллелизма. Но в процессе моделирования могут быть выявлены решения критических проблем в построении квантового компьютера и его программирования, кроме того – эффективное моделирование может позволить решать некоторые задачи (например, задачи управления или поиска) не дожидаясь появления квантового компьютера, так с помощью некоторых способов моделирования за приемлемое время можно выполнять алгоритм Шора на персональном компьютере с входными данными до 100 кубитов, алгоритм Гровера поиска в неупорядоченной базе данных – более 1000 кубитов.

Список литературы

1. Ulyanov S.V., Litvintseva L.V., Ulyanov S.S. Quantum information and quantum computational intelligence: Design & classical simulation of quantum algorithm gates. – Universita degli Studi di Milano: Polo Didattico e di Ricerca di Crema Publ, 2005. – Vol. 80.
2. Ulyanov S.V. System and method for control using quantum soft computing // US patent. – 2003. – № 6 578 018 B1.
3. Benenti G., Casati G., Strini G. Principles of quantum computation and information. – Singapore: World Scientific, 2004. – Vol. 1.
4. Китаев А., Шень А., Вялый М. Классические и квантовые вычисления. – М: Прогресс, 1999.
5. Прескилл Дж. Квантовая информация и квантовые вычисления. – Ижевск: Научно-издательский центр «Регулярная и хаотическая динамика», 2008. – Т. 1.
6. Риффель Э., Полак В. Основы квантовых вычислений // Квантовые компьютеры и квантовые вычисления: перевод с англ.: Романюк А.Ю., Федичкин Л.Е. – 2000. – №1.
7. Ulyanov S.V. Efficient simulation system of quantum algorithm gates on classical computer based on fast algorithm // Systemics, Cybernetics and Informatics. – 2004. – Vol. 2. – № 3. – Pp. 63-68.