Московский государственный университет им. М.В. Ломоносова,
физический факультет, кафедра общей физики и волновых процессов

Исследование возможностей реализации квантовых алгоритмов на аналоговых вычислительных машинах

Курсовая работа
студента 426 группы
Сыча Дениса Васильевича

Научный руководитель
к. ф.-м. н., доцент
Борис Андреевич Гришанин

Москва, 15 мая 2000 г.

Содержание

1 Исторический обзор

Впервые идея квантовых вычислений была высказана в 1982 году Ричардом Фейнманом [1]. В своей статье он обсуждал два вопроса. Во-первых, существуют ли какие-нибудь физические ограничения (типа начал термодинамики) на функционирование компьютера, накладывающие запреты на реализуемость алгоритмов (запрет на существование вечного двигателя первого и второго рода)? Оказалось, что таких энергетических ограничений нет. Если мы организуем вычисления обратимым образом, то затраты энергии будут сколь угодно малые. В реальном компьютере идут необратимые вычисления, но их можно сделать обратимыми без заметной потери эффективности. Кроме того, энергетический эффект необратимости вычислений в современных компьютерах пренебрежимо мал по сравнению с тепловыми шумами. Таким образом, нет принципиальных барьеров для быстрого решения сложных задач. Второе обстоятельство, отмеченное Фейнманом — это то, что если у нас есть квантовое вычислительное устройство, т.е. подчиняющееся законам квантовой механики, то его возможности могут превосходить возможности обычных классических вычислительных устройств. Тогда еще было непонятно, в каких конкретно задачах можно использовать квантовый компьютер (далее КК). Фейнман предположил, что КК может быть полезен для моделирования самих квантовых систем.

До Фейнмана в 1980 г. Беннефф [2] показал, что обратимая унитарная эволюция в состоянии реализовать Машину Тьюринга, т.е. вычислительная мощность КК не меньше, чем у классического. Но он не выяснял, являются ли квантовые устройства более мощными, чем классические. Да и сам термин *квантовый компьютер* он не употреблял. Дойч [3] в 1985 г. был первым, кто создал формальную теорию квантовых вычислений. Он определил квантовую Машину Тьюринга и квантовую цепь, а также некоторые свойства этих систем.

Вопрос об увеличении мощности КК по сравнению с классическим компьютером был рассмотрен в 1992 году Дойчем и Джозей [4] а также Берзьямом и Брассардом [5]. В этих работах было показано, что имеются задачи, эффективно решаемые на КК (асимптотически быстрее, чем на классическом). Но примеры этих задач несколько искусственные, не имеющие практического значения. Например, Дойч придумал такую задачу: для булевой функции определить, выполняется ли соотношение $f(0) = f(1)$ или нет. Если на классическом компьютере для этого требуется вычислить функцию два раза: $f(0)$ и $f(1)$, то на квантовом — один раз. Это достигается за счет того, что квантовый компьютер оперирует не с 0 и 1 а с суперпозицией этих чисел. Этот достаточно простой пример показывает принципиальные преимущества КК.

В статье Бернштейна и Вазирани [6] 1993 года и Саймона [7] 1994 года было предложено решение *проблемы оракула*, занимающее полиномиальное время на КК (на классическом компьютере требуется экспоненциальное время). Проблема оракула в одной из формулировок заключается в следующем. У нас есть некоторая функция $f : \{0,1\}^n \to \{0,1\}^n$. Устройство этой функции (формулу) мы не знаем (для нас это черный ящик), но она мгновенно вычисляет свое значение (поэтому функция и называется *оракул*). Конечно, физически

это нереально, но здесь под мгновенностью понимается то, что время вычисления оракула не учитывается в общем времени работы алгоритма. Кроме того, известно, что эта функция имеет "период"a, т.е. $f(x \oplus a) = f(x)$ (символом $\oplus$ обозначена побитная операция XOR). Необходимо найти этот “период". Очевидно, что для классического компьютера это экспоненциально сложная (относительно n) задача, а для КК, как уже было сказано выше, можно найти решение за полиномиальное число шагов.

Шор [8] в 1994 г. придумал полиномиальный по времени квантовый алгоритм разложения числа на простые множители, что уже имело большое практическое значение. Гровер [9] в 1995 г. предложил алгоритм поиска элемента в несортированной базе данных из N элементов за число шагов порядка $\sqrt{N}$. Китаев [10] в 1995 г. нашел решение одной математической проблемы, в частном случае приводящей к еще одному алгоритму факторизации чисел, также занимающему полиномиальное время.

Еще одно важное направление теоретического исследования квантовых компьютеров — это коррекция ошибок. Во-первых, любая физическая реализация КК будет приближенной. Во-вторых, работа любого устройства подвержена всяким случайным (спонтанным) ошибкам. Для обычных компьютеров эти проблемы довольно просто решаются введением избыточной информации и применением специальных экономных кодов. В КК эта проблема гораздо глубже. В придачу к “классическим"проблемам в КК имеются еще свои специфические “квантовые"проблемы. Например проблема декогеренции (распад суперпозиционного квантового состояния на классические) и неклонируемости квантовой информации (в КК нельзя просто взять и сделать копию информации, скопировать один бит в другой). В этом направлении также сделаны важные шаги. Итоговый результат состоит в том, что имеется некоторый максимально допустимый относительный размер ошибки ($\approx 10^{-6}$), до которого ошибки корректируются, а если ошибка больше — то нет. Таким образом, достаточно "аккуратные"вычисления на КК можно проводить неограниченно долго. Далее мы не будем обсуждать вопрос коррекции ошибок, обзорная статья на эту тему есть, например, у Прескилла [11].

2 Цели работы

Данная работа состоит из трех логических частей. В первой части (раздел 3) дано краткое введение в теорию квантовых вычислений. Во второй части (раздел 4) рассматривается возможность реализации квантовых алгоритмов на "неквантовых"физических системах (на классическом аналоговом компьютере), а в третьей (раздел 5) предлагается способ реализации произвольных квантовых алгоритмов на двукубитном КК. Ожидание положительного результата основано на некоторых теоретических результатах моделирования КК на *Машине Тьюринга* (далее МТ). Формальное определение МТ [14] здесь не целесообразно, поэтому под МТ сейчас можно понимать обычный компьютер типа IBM PC.

Для задачи на КК существует три типа ресурсов: память, время и точность. Память КК это число используемых кубитов, время — число шагов в данном алгоритме и точность это

точность с которой кубит воспринимается данной физической реализацией КК. При непосредственном моделировании КК на МТ “в лоб"(представление квантовой системы на классическом компьютере) необходимо экспоненциальное увеличение всех этих ресурсов. В работе Bernstein and Vazirani [6] было показано, что можно моделировать КК на МТ, используя только полиномиально большую память. Таким образом, необходимо экспоненциальное увеличение не всех ресурсов МТ по сравнению с КК. Видимо, можно моделировать работу КК с экспоненциальным увеличением только одного из ресурсов КК (память, время и точность). В качестве такого ресурса предлагается выбрать точность. При этом “внешние"физические параметры вычислительного устройства (память — размер и время — число шагов) останутся по крайней мере (в худшем случае) полиномиально большими. Кроме того, недавно появилась работа Ораевского [12], в которой утверждается принципиальная возможность моделирования КК на классических системах типа электромагнитного поля в лазерном резонаторе.

Раздел 5 посвящен реализации “больших"квантовых алгоритмов на “маленьких"КК. Экспериментально созданы небольшие КК с числом кубитов меньше 10. Для практического применения КК, например для факторизации больших чисел, необходимо около тысячи кубитов. Вопрос состоит в том, как использовать экспериментально созданные КК для таких задач. Показывается, что любой квантовый алгоритм можно реализовать на двукубитном компьютере с полиномиальной задержкой по времени (в худшем случае с квадратичной задержкой N^2, где N — число кубитов в “большом"КК). Кроме самого двукубитного квантового компьютера необходимо также устройство ввода-вывода квантовой информации и внешняя память, способная хранить квантовую информацию .

3 Основные понятия теории квантовых вычислений

3.1 Кубит

Главным элементом квантового компьютера является квантовая двухуровневая система — квантовый бит, или просто *кубит* (от quantum bit of information, qubit). Квантовая природа кубита заключается в принципе суперпозиции, согласно которому кубит находится одновременно сразу в обоих своих базовых состояниях. Для записи базовых функций состояния, соответствующих классическим состояниям 0 и 1, будем использовать обозначения Дирака: $|0\rangle$ и $|1\rangle$. Итак, состояние кубита запишем в виде $|q\rangle = a\,|0\rangle + b\,|1\rangle$. При измерении мы обнаружим кубит в состоянии $|0\rangle$ с вероятностью $|a|^2$ и в состоянии $|1\rangle$ с вероятностью $|b|^2$. Считаем, что других состояний у кубита нет, поэтому выполняется условие нормировки: $|a|^2 + |b|^2 = 1$. Также считаем, что кубит с течением времени самопроизвольно не изменяет своего состояния, и никаких нежелательных (не запланированных алгоритмом) действий с кубитом не происходит.

3.2 Квантовый регистр

Набор N кубитов составляет *квантовый регистр*, или просто регистр. Принципиальным моментом теории является то, что этот регистр, вообще говоря, также подчиняется принципу суперпозиции и находится одновременно сразу во всех своих базовых классических состояниях, число которых равно $Q = 2^N$. Произвольное состояние регистра записывается в виде: $|R\rangle = \sum_{i=1}^{Q} r_i |i\rangle$. Базисное состояние регистра $|i\rangle$ понимается в следующем смысле: число i в двоичной записи показывает состояния составляющих регистр кубитов. Например, мы говорим, что трехкубитный регистр находится в состоянии $|6\rangle$, или в двоичной записи $|6 = 110_2\rangle$, когда составляющие регистр кубиты имеют состояния: $q_1 = |1\rangle$, $q_2 = |1\rangle$, $q_3 = |0\rangle$. Коэффициенты r_i имеют такой же вероятностный смысл, как и у кубита: $|r_i|^2$ — это вероятность обнаружить регистр в состоянии $|i\rangle$. Также выполняется условие нормировки: $\sum_{i=1}^{Q} |r_i|^2 = 1$. Пример квантового регистра из трех кубитов: $|R\rangle = \frac{1}{\sqrt{2}} |010\rangle + \frac{i}{2} |001\rangle + \frac{1}{2} |110\rangle$. Производя вычисления с квантовым регистром для выяснения каких-то общих свойств функции, на выходе из квантового вычислительного устройства мы имеем суперпозицию 2^N функций от всех классических состояний регистра. Важным моментом является тот факт, что произвольное состояние регистра, вообще говоря, не является произведением состояний отдельных кубитов типа $|R\rangle = |q_1\rangle |q_2\rangle ... |q_N\rangle$. Например, в состоянии регистра $|R\rangle = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle$ для каждого кубита невозможно указать его состояние в виде $|q\rangle = a |0\rangle + b |1\rangle$, т.е. состояние регистра не является тензорным произведением состояний кубитов. Про такое состояние кубита говорят, что оно *перепутано* (entangled) с состояниями других кубитов. Перепутанные состояния регистра, собственно говоря, это как раз то место, где и возникает новое качество КК по сравнению с классическим. Если бы его не было, то это был бы просто *вероятностный компьютер*. Модель вероятностного компьютера (ВК) далее в работе рассматриваться не будет, хотя это тоже важное понятие из теории вычислений. Коротко ВК можно описать так: вычисления на следующем шаге не строго детерминированы предыдущим состоянием компьютера, а определены лишь вероятностным образом. Существуют *вероятностные алгоритмы*, которые выполняются на ВК гораздо эффективней, чем на обычном компьютере. Итак, состояние регистра не может быть описано простым произведением состояний каждого кубита, а является экспоненциально большой суперпозицией всех своих базовых состояний.

3.3 Гейты и квантовая логика

Любая логическая операция с кубитами называется *гейтом* (от английского gate – ворота). Название видимо обусловлено тем, что по ходу алгоритма кубиты как бы "проходят"сквозь гейты, при этом изменяя свое состояние. По числу задействованных кубитов гейты делятся на одно- и многокубитные. Гейт переводит одно состояние регистра в другое. Физически реализуемые гейты соответствуют унитарным операциям с волновой функцией регистра. Действие гейта на регистр можно записать так: $G |R\rangle = |R'\rangle$. Гейты — линейные операции:

$G(|p\rangle + |q\rangle) = G\,|p\rangle + G\,|q\rangle)$. Произвольный пример действия гейта:

$$G(\frac{i}{\sqrt{2}}\,|010\rangle + \frac{1}{2}\,|001\rangle + \frac{i}{2}\,|110\rangle) = \frac{1}{2}\,|010\rangle + \frac{1}{2}\,|101\rangle + \frac{1}{\sqrt{2}}\,|111\rangle\,.$$

Для демонстрации действия гейта на кубиты используют матричную запись гейта или же таблицу истинности. Матрица гейта действует на столбец весовых коэффициентов регистра и получается новый столбец, соответствующий новому состоянию регистра. В случае, если в действии гейта не участвуют некоторые кубиты, то их и не включают в матрицу, т.е. в матрице записано только реальное действие кубитов. Пример матричной записи кубита будет дан ниже. Таблица истинности отражает действие гейта на базисные состояния. Ее структура имеет следующий вид: по горизонтали записывается слева начальные состояния входящих кубитов, а справа — соответствующие конечные. По вертикали записываются все базисные состояния.

В теории классических вычислений показано, что любую логическую операцию можно представить как совокупность некоторого числа стандартных, базовых операций. В качестве такого базиса логических операций можно взять пару NOT и XOR (XOR иногда называют CNOT– контролируемый NOT). В квантовом случае имеет место аналогичная ситуация: любую обратимую унитарную операцию на кубитах можно представить как совокупность базовых операций. Базисом квантовой логики может служить один трехкубитный гейт (например Тоффоли (CCNOT) или Фредкина (CSWAP) — описание этих гейтов будет дано ниже) или один однобитный и один двубитный гейт (например NOT и CNOT). Именно последний вариант базиса — NOT и CNOT мы и будем рассматривать в дальнейшем.

Однобитная логическая операция NOT переводит $|q\rangle = a\,|0\rangle + b\,|1\rangle$ в $|q'\rangle = b\,|0\rangle + a\,|1\rangle$, т.е. переставляет весовые коэффициенты кубита местами. В классическом случае ей соответствует обычный NOT, т.к. один из коэффициентов равен нулю.

Двубитный гейт CNOT (Controlled NOT), действующий на двукубитное состояние в общем виде записывается так:

$$CNOT(R_{00}\,|00\rangle + R_{01}\,|01\rangle + R_{10}\,|10\rangle + R_{11}\,|11\rangle) = R_{00}\,|00\rangle + R_{01}\,|01\rangle + R_{11}\,|10\rangle + R_{10}\,|11\rangle$$

В классическом случае это просто XOR.

Для пояснения запишем матрицу действия этих гейтов:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}; \quad NOT = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Таблица истинности для них выглядит так:

$$CNOT: \begin{array}{|cc|cc|} \hline 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ \hline \end{array}; \quad NOT: \begin{array}{|c|c|} \hline 0 & 1 \\ 1 & 0 \\ \hline \end{array}.$$

Иногда используется графическая форма записи квантовых алгоритмов. Гейты обозначаются некоторыми символами (часто это кружок или квадрат с цифрой или буквой внутри). Кубиты представлены горизонтальными нитями. Действие гейта на кубит показывается путем "нанизывания"гейта на нужный кубит (или несколько кубитов, если это не однобитный гейт). Квантовый алгоритм представляется в виде сети таких гейтов и называется квантовой сетью. Слева в такой сети находятся начальные состояния кубитов, справа — конечные. Действие алгоритма заключается в прохождении кубитов по своим нитям через гейты слева направо. Среди гейтов важное место занимают *контролируемые* или *условные* гейты. Это такие гейты, которые изменяют один из кубитов — кубит цели, при условии равенства 1 управляющего (контролирующего) кубита. Для таких гейтов используется стандартное обозначение: гейт "нанизывается"на изменяемый кубит соответствующим символом, а к управляющему кубиту, или нескольким кубитам, "цепляется"вертикальной нитью с черной точкой на конце. В символьной записи эти гейты записываются в виде CGate, где Gate — это операция с кубитом цели. Операция NOT графически выглядит как крест в кружке — $\oplus$.

3.4 Описание используемых гейтов

Кроме упомянутых выше гейтов NOT и CNOT в квантовых вычислениях используются также некоторые другие гейты. Их применение не необходимо, как уже было сказано выше, но запись алгоритма с их помощью намного проще. На практике часто используются такие гейты: однобитный H (Hadamard), двубитные R (rotate), S (swap), трехбитные CCNOT (гейт Тоффоли), CSWAP (гейт Фредкина). Действие гейтов показывается на базисных (классических) состояниях, в силу принципа суперпозиции и линейности гейтов, распространяется и на произвольные состояния регистра.

$$H\,|x\rangle = \frac{1}{\sqrt{2}}(-1)^{xy}\,|y\rangle \quad R_{k,l}(|k\rangle,|l\rangle) = (|k\rangle,(\overline{k}\otimes 1 + k\otimes e^{i\pi 2^{k-l}})\,|l\rangle); \quad \mathrm{S}(|x\rangle,|y\rangle) = (|y\rangle,|x\rangle);$$

$$\mathrm{CCNOT}(|x\rangle,|y\rangle,|z\rangle) = (|x\rangle,|y\rangle,|x\otimes y\oplus z\rangle);$$

$$\mathrm{CSWAP}(|x\rangle,|y\rangle,|z\rangle) = (|x\rangle,|x\otimes z + \overline{x}\otimes y\rangle,|x\otimes y + \overline{x}\otimes z\rangle).$$

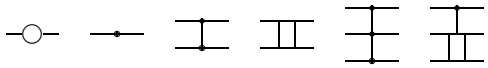
Знак $\otimes$ означает логическое умножение, знак $\oplus$ — сложение по модулю 2 (логическая операция CNOT): $a\oplus b = a+b\ mod\ 2$.

Матрицы гейтов:

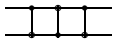
$$\mathrm{H}: \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}; \quad \mathrm{S}: \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \quad R_{k,l}: \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\pi 2^{k-l}} \end{pmatrix};$$

$$\text{CCNOT}: \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \quad \text{CSWAP}: \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Графическое изображение гейтов приведено ниже (слева направо — гейт Хадамара, NOT, CNOT, SWAP, CCNOT, CSWAP):



Гейт SWAP можно реализовать на трех гейтах CNOT следующим образом:



Это несложно проверить для базисных состояний: $|0\rangle|1\rangle \to |0\rangle|1\rangle \to |1\rangle|1\rangle \to |1\rangle|0\rangle$; $|1\rangle|0\rangle \to |1\rangle|1\rangle \to |0\rangle|1\rangle \to |0\rangle|1\rangle$; $|1\rangle|1\rangle \to |1\rangle|0\rangle \to |1\rangle|0\rangle \to |1\rangle|1\rangle$; $|0\rangle|0\rangle \to |0\rangle|0\rangle \to |0\rangle|0\rangle \to |0\rangle|0\rangle$, а в силу принципа суперпозиции это справедливо и для любых состояний.

3.5 Общий вид квантовых алгоритмов

Идея о суперпозиционном состоянии квантового регистра наводит на мысль о возможности использования квантовых вычислений в задачах, где необходим перебор входных данных для отыскания элементов, удовлетворяющих некоторым условиям. Например, это может быть задача о нахождении уникального элемента на некотором множестве (функция, проверяющая “уникальность"равна 1 на данном элементе и 0 для всех остальных элементов). Использование классического перебора приводит к числу действий порядка N (N — число элементов во множестве). Квантовая схема (алгоритм Гровера) находит ответ за число операций порядка $\sqrt{N}$. Другая важная задача, эффективно решаемая на квантовом компьютере, это факторизация чисел (факторизация — разложение числа на простые множители). Пусть число состоит из N десятичных знаков. Самый лучший классический алгоритм факторизации требует порядка $e^{\sqrt[3]{N}}$ действий. В таком случае говорят, что алгоритм имеет экспоненциальную сложность, или время вычислений экспоненциально. Квантовый алгоритм (т.е. алгоритм, использующий преимущества квантового регистра) справится с этой задачей за полиномиальное время (алгоритм факторизации Шора и альтернативный алгоритм Китаева). Общая структура квантовых алгоритмов примерно такова: сначала создается равномерная суперпозиция всех возможных классических состояний регистра, далее идут некоторые преобразования регистра, в результате чего некоторые состояния приобретают гораздо большую вероятность,

чем остальные. Это возможно благодаря эффекту интерференции квантовых состояний, который заключается в том, что в зависимости от фазы кубита, вычисления с ним приводят к разным конечным состояниям. Поясним сказанное на следующем примере. Рассмотрим гейт CH (Controlled Hadamard), описываемый следующей матрицей:

$$CH = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}.$$

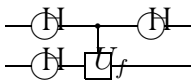
Предположим, что в начале регистр находился в состоянии $|R_1\rangle = \frac{1}{\sqrt{2}}|10\rangle - \frac{1}{\sqrt{2}}|11\rangle$. Под действием гейта CH он перейдет в состояние $|11\rangle$. В начале вероятность обнаружить регистр в состоянии $|11\rangle$ была равна $\frac{1}{2}$. Теперь эта вероятность равна 1. Теперь изменим фазу у второй части начальной суперпозиции регистра на π. Имеем $|R_2\rangle = \frac{1}{\sqrt{2}}|10\rangle + \frac{1}{\sqrt{2}}|11\rangle$. Под действием того же гейта получим состояние $|10\rangle$, т.е. совершенно другой ответ. Таким образом, вычисления с квантовым регистром могут привести к некоторым выделенным состояниям. При измерении регистра мы и получаем эти состояния. Процесс измерения приводит к разрушению квантовой суперпозиции, и в результате одного измерения мы обнаружим только одно состояние. Производя квантовые вычисления несколько раз мы получим распределение вероятностей состояний регистра. Квантовые алгоритмы устроены таким образом, что эти вероятности сильно превосходят вероятности остальных состояний.

Таким образом сокращается множество потенциально правильных ответов для данной задачи. Далее идет обработка результатов классическим компьютером.

3.6 Пример квантового алгоритма — задача Дойча

Для демонстрации введенных понятий разберем один достаточно простой квантовый алгоритм. Задача Дойча, как уже было сказано, заключается в выяснении свойства сбалансированности булевой функции. Функция называется *сбалансированной*, если $f(0) \neq f(1)$ и *постоянной* в противном случае. Ясно, что на классическом компьютере необходимо дважды вычислить данную функцию. Только после этого мы можем ответить на поставленный вопрос. Рассмотрим возможности квантового компьютера для решения этой задачи. Пусть у нас есть унитарное преобразование двухкубитного регистра вида: $U_f : |x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$, которое инвертирует второй бит, при условии равенства $f(x) = 1$. Если в $|x\rangle|y\rangle$ записана классическая информация в виде 0 или 1, то нам, очевидно, опять придется вычислять функцию дважды. Поэтому на вход подается квантовая суперпозиция 0 и 1: $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$, которая подготавливается применением гейта H к соответствующим классическим состояниям. Действие гейта H на базисные состояния: $|0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad |1\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

Квантовая цепь, реализующая алгоритм Дойча:



Последовательность операций над кубитами в сети:

$$|0\rangle\,|1\rangle \to \frac{1}{2}(|0\rangle+|1\rangle)(|0\rangle-|1\rangle) \to \frac{1}{2}\left((-1)^{f(0)}\,|0\rangle+(-1)^{f(1)}\,|1\rangle\right)(|0\rangle-|1\rangle) \to$$

$$\frac{1}{2}\left[\left((-1)^{f(0)}+(-1)^{f(1)}\right)|0\rangle+\left((-1)^{f(0)}-(-1)^{f(1)}\right)|1\rangle\right]\frac{1}{\sqrt{2}}(|0\rangle-|1\rangle)$$

Итак, на вход цепи подаются чисто классические состояния $|0\rangle\ |1\rangle$. Гейты H переводит их в соответствующее суперпозиционное состояние. Далее действует гейт U_f, после чего задача уже, собственно говоря, решена. Первый кубит в результате действия этого гейта переходит в одно из состояний $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$ в зависимости от свойств функции $f(x)$. Проектированием первого гейта на базис $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$ мы бы узнали ответ. Но можно еще раз применить гейт H к первому кубиту, как это здесь и сделано, переводя его в классическое состояние $|0\rangle\ |1\rangle$. Заканчивается алгоритм измерением состояния первого кубита. Если его значение равно 0, то функция постоянна: $f(0) = f(1)$ (т.к. тогда $(-1)^{f(0)} - (-1)^{f(1)} = 0$), если же он равен 1, то $f(0) \neq f(1)$. Таким образом квантовый алгоритм решает залачу эффективней классического.

4 Моделирование КК на аналоговой технике

4.1 Об обозначениях сложности алгоритмов

Для классификации алгоритмов вводится понятие сложности алгоритма. Иногда говорят также о сложности задачи, при этом понимают сложность наилучшего алгоритма для решения этой задачи. Также задачу иногда называют проблемой. В классических вычислениях существует два различных класса сложности: P и NP (на самом деле их больше, если брать в расчет ВК, но здесь другие классы нам не понадобятся). Задача называется P-сложной или полиномиальной (Polynomial), если она решается за полиномиальное время, т.е. число действий будет порядка некоторой степени от размера входных данных. Задача называется NP-сложной (Nondeterministic Polynomial) или суперполиномиальной (можно считать ее экспоненциальной, хотя это и не совсем так) если для ее решения нет полиномиального алгоритма. Для полного решения NP-задачи на классическом компьютере необходимо число действий, в асимптотике большее любой степени от размера входных данных (суперполиномиальное время). NP-проблемы в свою очередь делятся на NP-полные и NP-неполные. Если проверка какого-то ответа на истинность является P-проблемой, то задача называется NP-неполной. Пример такой задачи — факторизация чисел. Разложить число на простые множители сложно, но проверить, является ли данный ответ правильным, легко. Если же проверка ответа на истинность не является P-проблемой, а более сложна, то задача называется NP-полной. Пример NP-полной задачи — известная *задача коммивояжера*, или по-английски

TSP (Traveling Salesman Problem). Математическая формулировка этой задачи: на заданном графе найти кратчайший путь между двумя точками. Если даже кто-то "подскажет"нам правильный ответ, то мы не сможем убедится в его истинности за полиномиальное время.

Если замедлить или ускорить алгоритм в полиномиальное число раз (не большее некоторой степени от размера входных данных), то сложность алгоритма не изменится — P-проблема останется P, а NP — NP. На заре теории вычислений было показано [14], что любая задача, решаемая на классической вычислительной технике может быть решена на МТ без изменения сложности, т.е. промоделирована на МТ максимум с полиномиальной задержкой. Поэтому модель Машины Тьюринга является базовой для теоретических рассмотрений алгоритмов. Основное предположение теории сложности заключается в том, что $P \neq NP$, поэтому P/NP критерий позволяет качественно оценить асимптотическую эффективность алгоритма.

4.2 Моделирование КК на Машине Тьюринга

Для того, чтобы моделировать КК на обычном компьютере, нам необходимо хранить информацию о весовых коэффициентах всех состояний регистра. Это приводит к экспоненциальному росту памяти машины (по отношению к КК). Кроме того, производя изменения состояний некоторых кубитов, мы должны изменить состояния экспоненциально большого числа коэффициентов в регистре, и это приводит к экспоненциальному росту числа операций, т.е. времени. Однако было показано [13], что можно моделировать КК на МТ на полиномиальном пространстве, но время при этом остается экспоненциальным.

4.3 О классическом представлении кубита

Рассмотрим один кубит $|q\rangle = a\,|0\rangle + b\,|1\rangle$. Здесь a и b — комплексные числа , связанные соотношением нормировки $|a|^2 + |b|^2 = 1$. Каждое из них имеет действительную и мнимую составляющие — итого кубит можно представить как совокупность четырех действительных чисел: $|q\rangle = (q_1, q_2, q_3, q_4)$. Назовем эту совокупность *классической проекцией* кубита. Ей можно поставить в соответствие кубит так: $|q\rangle = (q_1 + iq_2)\,|0\rangle + (q_3 + iq_4)\,|1\rangle$. Соотношение нормировки $\sum_{i=1}^{4} q_i^2 = 1$ показывает ограниченность каждого числа: $|q_i| \leq 1$. Итак, кубит можно мыслить как вектор в четырехмерном пространстве, а гейт — как вращение этого вектора. Более того, четыре действительных ограниченных числа можно мыслить как одно число, в котором они по определенному правилу закодированы. Это утверждение можно распространить и на более общий случай, воспользовавшись тем, что мощность счетного числа действительных чисел в теоретико-множественном смысле эквивалентна мощности одного действительного числа. В дальнейшем мы будем подразумевать этот факт, но говорить о нескольких числах. Реально в машине это может быть одно число. Изменение состояния кубита под действием гейта имеет взаимно–однозначное соответствие с изменением его классической проекции. Любому гейту будет соответствовать функция на пространстве классических проекций (*функциональное*

представление гейта). Производя вычисления с классическими проекциями кубитов, мы получаем соответствующее конечное состояние квантового регистра. Для того, чтобы узнать вероятности всех классических состояний этого регистра, нам опять было бы необходимо экспоненциальное время, но на самом деле нам достаточно знать лишь самые вероятные состояния регистра, поэтому экспоненциального замедления алгоритма не происходит. Такой подход применим для моделирования алгоритмов, не использующих свойства перепутанности, например *квантового преобразования Фурье* (см. ниже). Для корректного отображения перепутанности кубита нам недостаточно знать только одну четверку (q_1, q_2, q_3, q_4). В таком случае надо вводить дополнительные числа для отображения весовых коэффициентов всего регистра. Их число будет экспоненциально большим, но в аналоговой машине это может быть, как уже говорилось выше, одно число, задаваемое с экспоненциально большой точностью. Для демонстрации возможности такого представления приведем функциональное представление гейтов NOT и CNOT, как базисных элементов квантовой логики. Операции NOT будет соответствовать функция FuncNOT и CNOT — FuncCNOT.

$$NOT(a\,|0\rangle + b\,|1\rangle) = b\,|0\rangle + a\,|1\rangle$$
$$FuncNOT(q_1, q_2, q_3, q_4) = (q_3, q_4, q_1, q_2)$$
$$CNOT(R_{00}\,|00\rangle + R_{01}\,|01\rangle + R_{10}\,|10\rangle + R_{11}\,|11\rangle) =$$
$$R_{00}\,|00\rangle + R_{01}\,|01\rangle + R_{11}\,|10\rangle + R_{10}\,|11\rangle$$
$$FuncNOT(R_{00}, R_{01}, R_{10}, R_{11}) =$$
$$(R_{00}, R_{01}, R_{11}, R_{10})$$

5 Преобразование квантовых алгоритмов

5.1 Трудности квантовых вычислений

Для реализации преимуществ квантовых вычислений перед классическими, необходимы задачи, с которыми классические вычислительные машины не справляются. Речь идет не о принципиальной "нерешаемости"задачи, а о времени, необходимом им на это решение. Если оно превосходит разумный предел (скажем, один год), то задача считается неразрешимой. В 1994 был проведен эксперимент по факторизации 129-разрядного числа, называемого RSA129 [18]. С использованием 1600 рабочих станций, объединенных через Интернет, это удалось сделать за 8 месяцев. Таким образом для факторизации чисел предельная длина числа составляет примерно 150 десятичных знаков. В двоичной записи это приблизительно 400 знаков. Квантовый компьютер с таким регистром (400 кубитов) будет обладать принципиальными новыми вычислительными способностями, т.к. он справится с задачей факторизации за разумное время (конкретное время зависит от быстродействия элементной базы КК). Однако в настоящее время такие КК не созданы и это главное препятствие на пути реализации квантовых алгоритмов. Сейчас созданы "маленькие"КК (< 10 кубитов). Возникает вопрос: а нельзя ли эффективно использовать эти небольшие КК, что бы моделировать работу боль-

ших, полномасштабных КК?

5.2 Идея метода и пример

Во всех используемых в настоящее время алгоритмах применяются гейты, действующие на различные кубиты. В такой схеме вычислений каждая произвольная пара кубитов обязательно провзаимодействует (найдется гейт, использующий именно эту пару кубитов). В случае, если кубиты представлены пространственно удаленными системами (спины атомов) применять гейты, действующие на далеко удаленные друг от друга кубиты, может оказаться сложной задачей. Поэтому возникает вопрос о преобразовании квантовых алгоритмов к виду, когда используются только гейты, действующие на "соседние"кубиты. Под "соседними"понимаются кубиты с номерами, отличающимися на единицу. Такое преобразование любого алгоритма можно осуществить следующим образом: когда необходимо применить гейт на удаленные кубиты, мы "подводим"один из этих кубитов операцией SWAP к другому кубиту путем последовательного обмена состояниями с соседними кубитами, как в методе сортировки "пузырьком". После этого производим нужную нам операцию и "отводим"кубит таким же образом на прежнее место.

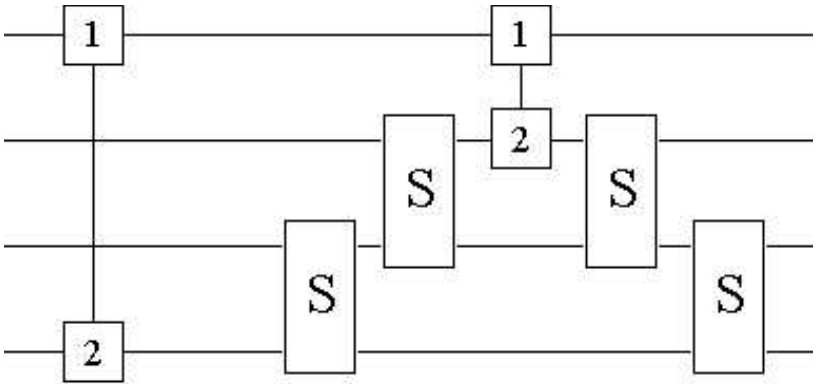


Рис. 1: Модификация произвольного гейта 1-2. Слева представлен произвольный двукубитный гейт 1-2, справа — его модификация к "короткодействующему"виду. Гейт S обозначет SWAP и описан выше (см. описание гейтов).

При этом алгоритм получит полиномиальную задержку порядка N^2, где N — размер входных данных (число кубитов). Но сложность алгоритма от этого, как уже обсуждалось, не изменится. Продемонстрируем сказанное на примере важного квантового алгоритма — квантового быстрого преобразования Фурье (QFFT – Quantum Fast Fourier Transform). Этот квантовый алгоритм участвует в решении всех практически интересных задач (факторизация и быстрый поиск). Пусть $Q = 2^n$. По определению, преобразованием Фурье базисного состоя-

ния регистра $|a\rangle$ называется новое состояние $|\Psi_{Q,a}\rangle$ такое, что:

$$|\Psi_{Q,a}\rangle \equiv \frac{1}{\sqrt{Q}} \sum_{b=0}^{Q-1} e^{i2\pi \frac{a}{Q} b} |b\rangle$$

Интересно, что в данном алгоритме состояния кубитов не перепутаны, и состояние регистра записывается в виде тензорного произведения состояний кубитов. Его можно записать в виде:

$$|\Psi_{Q,a}\rangle \equiv \frac{1}{\sqrt{Q}}(|0\rangle + e^{i2\pi 0.a_1} |1\rangle)(|0\rangle + e^{i2\pi 0.a_{n-1}a_n} |1\rangle)...(|0\rangle + e^{i2\pi 0.a_1...a_{n-1}a_n} |1\rangle)$$

где $a_1...a_n$ — двоичное представление регистра: $|a\rangle = |a_1...a_n\rangle$. Это можно увидеть, вычисляя по модулю 1 коэффициент в экспоненте при $|b\rangle$ в определении $|\Psi_{Q,a}\rangle$ (коэффициент вычисляем по модулю 1, т.к. $e^{i2\pi(x+N)} \equiv e^{i2\pi x}$, где $N -$ любое целое число): $2^{-n}ab = 2^{-n} \sum\limits_{i,j=1}^{n} a_i 2^{n-i} b_j 2^{n-j} =$ $\sum\limits_{j=1}^{n} 0.a_1a_2...a_n \cdot b_j 2^{n-j} = 0.a_n \cdot b_1 + 0.a_{n-1}a_n \cdot b_2 + ... + 0.a_1...a_{n-1}a_n \cdot b_n$

А это и есть коэффициент экспоненты при $|b\rangle$ в правой части. Такой вид алгоритма позволяет делать преобразование Фурье, используя всего два гейта: условный "поворачивающий"гейт $R_{l,k}$ и гейт Хадамарда H_i, уже рассмотренные выше. Здесь под индексами гейтов понимаются номера кубитов, на которые они действуют. $n -$ гейт, осуществляющий преобразование Фурье, записывается в виде:

$$F = H_{l-1}R_{l-2,l-1}H_{l-2}R_{l-3,l-1}R_{l-3,l-2}H_{l-3}R_{l-4,l-1}R_{l-4,l-2}R_{l-4,l-3}H_{l-4}...$$
$$...H_1R_{0,l-1}R_{0,l-2}...R_{0,1}H_0$$

В этом можно убедиться, непосредственно вычислив итоговое состояние каждого кубита [16],[8]. Заметим, что некоторые операции можно выполнять параллельно, так как они действуют на разные кубиты. Графическая схема преобразования Фурье с учетом параллельно выполняемых операций представлена на рисунке 2. При модификации алгоритма также сохраняются преимущества параллельности операций.

После выполнения такой последовательности гейтов нужно еще переставить кубиты в обратном порядке, или условиться считать их наоборот. Перестановка кубитов в обратном порядке приведена на рисунке 3.

Модифицированное преобразование Фурье записывается так:

$$F = H_{l-1}R_{l-2,l-1}H_{l-2}S_{l-1,l-2}R_{l-3,l-2}S_{l-1,l-2}R_{l-3,l-2}H_{l-3}...$$
$$...S_{1,2}R_{0,1}S_{1,2}R_{0,1}H_0$$

Графическая схема приведена на рисунке 4.

Итак, для реализации любого квантового алгоритма нам необходимо иметь всего лишь двукубитный КК, устройство ввода-вывода и хранилище квантовой информации. Используя двухкубитный КК мы производим "конструктивные"операции с кубитами ($R_{l,k}$, H_i а также любые другие двукубитные операции) только "внутри"него, а обмениваясь информацией с внешней "памятью"мы задействуем в вычислениях любой кубит. Все, что требуется от "памяти", — это "ничего не делать"с кубитами, хранить именно квантовую информацию в суперпозиционном виде.

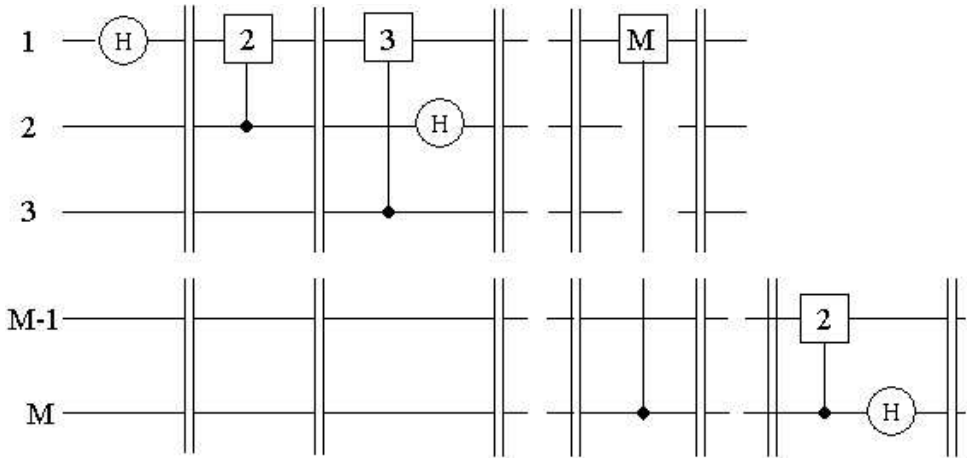


Рис. 2: Квантовая сеть, реализующая стандартное преобразование Фурье (см. также пояснение к рисункам).

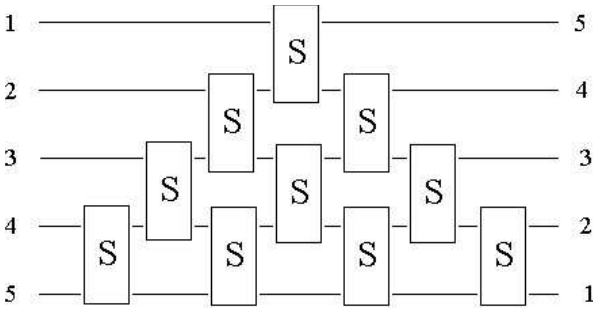


Рис. 3: Окончание квантового преобразования Фурье — перестановка кубитов в обратном порядке.

5.3 Пояснение к рисункам

Здесь изображены квантовые сети—массивы квантовых гейтов, соединенные горизонтальными нитями—кубитами. Слева—начальные состояния кубитов, справа—конечные.Каждый гейт действует на те биты, на которые он "нацеплен".Вертикальными двойными линиями разделены временные шаги.

Обозначения гейтов. Кружок с буквой H обозначает гейт Hadamard. Квадрат с цифрой i=l-k+1 внутри обозначает гейт Rotate $R_{l,k}$, вращающий фазу у $|\ 1>$ данного кубита (добавление 1 в i-ый регистр двоичной записи фазы) в зависимости от кубта, к которому он "прицеплен"вертикальной нитью с черной точкой на конце. Прямоугольник с буквой S обозначает гейт SWAP (переставляет кубиты местами).

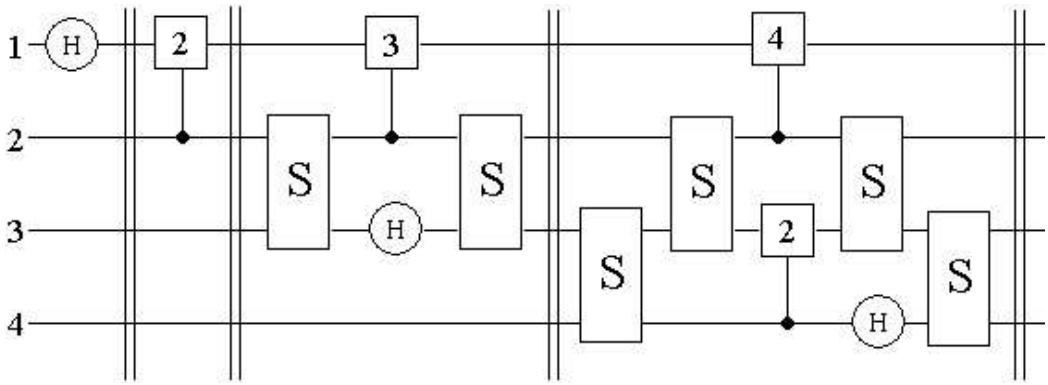


Рис. 4: квантовая сеть, реализующая модифицированное преобразование Фурье (начальный фрагмент)

6 Заключение

Итак, по результатам первой части работы можно сделать вывод о принципиальной возможности моделирования КК на классической аналоговой вычислительной технике без изменения сложности алгоритма. При таком подходе унитарным операциям будут соответствовать некие аналоговые микросхемы — оракулы. Время вычислений (число “шагов— микросхем) будет полиномиальным, но точность, необходимая для отображения числа внутри машины — экспоненциальной. Таким образом вполне реалистичным выглядит создание небольших классических эмуляторов КК.

Показано, что экспериментально созданные КК могут быть использованы для решения больших квантовых задач путем добавления устройства ввода-вывода квантовой информации. Для хранения квантовой информации могут быть использованы сами квантовые компьютеры (несколько дополнительных КК могут просто хранить информацию). [1]

[1]Для предварительного ознакомления с теорией квантовых вычислений рекомендуется следующая литература: Preskill [15], Aharonov [16], журнал "Квантовый компьютер"[17], Williams and Clearwater [18], Журнал "Компьютерра"[19], Лекции А. Китаева [20].

Список литературы

[1] R. Feynman, Simulating physics with computers, Internat. J. Theoret. Phys., 21 (1982), pp. 467-468.

[2] P. Benioff, The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines, J. Statist. Phys., 22 (1980), pp. 563-591.

[3] D. Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer, Proc. Roy. Soc. London Ser. A, 400, (1985) pp.96-117.

[4] D. Deutsch and R. Jossa, Rapid solution of problems by quantum computation, Proc. Roy Soc. London Ser. A, 439, (1992) pp.553-558.

[5] A. Berthiaume and G. Brassard, The quantum chalenge to structural complexity theory, in Proceedings of the Seventh Annual Structure in Complexity Theory Conference, IEEE Computer Society Press, Los Alamos, CA, (1992) pp. 132-137.

[6] E. Bernstein and U. Vazirani, Quantum complexity theory, in Proceedings of the 25th Annual ACM Symposium on Theory of Computing, ACM, New York, (1993) pp. 11-20.

[7] D. Simon, On the power of quantum computation, in Proceeding of the 35th Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, (1994) pp. 116-123.

[8] P. W. Shor, Algorithms for quantum computation: Discrete logarithms and factoring,in Proceeding of the 35th Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, (1994) pp. 124-134.

[9] L. K. Grover, Quantum mechanics helps in searching for a needle in a haystack, Phys. Rev. Lett. 79, 325-328 1997

[10] A. Yu. Kitaev, Quantum measurements and the Abelian stabilizer problem (1995), in LANL e-print quant-ph/9511026, http://xxx.lanl.gov

[11] J. Preskill, Reliable Quantum Computers (1997), in LANL e-print archive quant-ph/9705031, http://ru.arxiv.org/abs/quant-ph/9705031

[12] А.Н. Ораевский, "О квантовых компьютерах", Квантовая Электроника, том 30, N 5 (2000)

[13] E. Bernstein and U. Vazirani (1993), Quantum Complexity Theory, SIAM Journal of Computation 26 5 pp. 1411-1473 October, 1997

[14] A. Turing, "On Computable Numbers with an Application to the Entscheidungsproblem", Proceedings of the London Mathematical Society, Vol. 42(1937), pp. 230-265, erratum in 43(1937), pp. 544-546.

[15] Курс лекций в Интернете: http://www.theory.caltech.edu/ preskill/ph229

[16] D. Aharonov, "Quantum Computation"(1998), in LANL e-print archive quant-ph/9812037, http://ru.arxiv.org/abs/quant-ph/9812037

[17] Журнал "Квантовый Компьютер", адрес в Интернете: http://web.uni.udm.ru/ rcd/qc/index.html

[18] C.P. Williams S.H. Clearwater, "Explorations in Quantum Computing", ISBN 0-387-94768-X Springer-Verlag New York Berlin Heidelberg (1998).

[19] Интервью с М.Вялым, журнал "Компьютерра", N47(1997), http://www.computerra.ru/1997/47/3.html?inside

[20] Конспекты лекций А.Китаева в Московском Независимом Университете (записаны М.Вялым), адрес в Интернете: http://www.mccme.ru/ium/s98/quantcomp.html.