Классические и квантовые ветвящиеся программы

Аблаев Ф.М. Васильев А.В.

Оглавление

Глава 1

Вычисления в модели ветвящихся программ

1.1 Вычислительные задачи, языки и булевы функции.

В теоретической и прикладной кибернетике широко используется аппарат теории языков и булевых функций. В теории сложности языки и булевы функции используются также для представления вычислительных задач. Это позволяет иметь единое представление задач, что важно при анализе их сравнительных сложностных характеристик.

Определение 1.1.1. Алфавит — конечное множество символов.

Определение 1.1.2. Слово — конкатенация символов алфавита. Длина слова — число образующих его букв. e — пустое слово (длины 0).

X^* — множество всех слов конечной длины в алфавите X. X^n — множество всех слов длины n в алфавите X.

Определение 1.1.3. Язык — произвольное подмножество множества X^*, обозначаем L, $L \subseteq X^*$.

Определение 1.1.4. Для алфавита X функция $f : X^n \to \{0,1\}$ называется дискретной. Функция $f : \{0,1\}^n \to \{0,1\}$ называется булевой.

Содержательно представление задач в в виде языка и последовательности булевых функций можно описать следующим образом. Вычислительную задачу Z можно представить в виде слова в некотором выбранном алфавите X:
код(задачи Z) = код(исходных условий задачи Z) + код(исходных данных задачи Z) + код(ответа задачи Z). Таким образом, задача определяет некоторый язык:

$L_Z = \{w \in X^* : w =$ код(задачи Z) с ответом, соответствующим исходным условиям и данным $\}$

Языку $L \in X^*$ естественным образом можно сопоставить последовательность характеристических булевых функций $\mathcal{F}_L = \{f_0, f_1, \ldots, f_n, \ldots\}$, где $f_n : X^n \to \{0,1\}$ — дискретная функция такая, что

$$f_n(w) = \begin{cases} 1, & \text{если } w \in L \\ 0, & \text{если } w \notin L \end{cases}$$

1.2 Модели вычислений.

Под моделями вычислений (см., например, книгу [8]) понимаются контактные схемы, схемы из функциональных элементов, автоматы, машины Тьюринга и т.п.

Математические модели вычислений условно делятся на модели с памятью и модели без памяти. К первым относятся так называемые машинные, ко вторым – схемные модели вычислений. Между этими двумя классами моделей нет четких различий. В ряде случаев схемные модели можно непосредственно интерпретировать с машинной точки зрения. Типичной такой схемной моделью вычислений являются ветвящиеся (бинарные) программы. С одной стороны, ветвящаяся программа – это контактная схема, с другой стороны, в терминах ветвящихся программ естественным образом описываются машины Тьюринга и автоматы. В данной работе мы рассматриваем сравнительные возможности классических и квантовых моделей ветвящихся программ и конечных автоматов с машинной точки зрения.

Содержательно, вычислительная модель называется *однородной*, если один и тот же алгоритм работает на входах *различной длины*. *Семейство* алгоритмов называется *однородным*, если для входа любой длины подходящий алгоритм может быть просто сгенерирован машиной Тьюринга. В противном случае вычислительная модель называется неоднородной. Формализацию данного понятия смотри, например, в [31], [29].

К однородным вычислительным моделям относятся, например, машины Тьюринга, классические модели конечных автоматов. Схемы из функциональных элементов, ветвящиеся программы являются неоднородными.

1.3 Меры сложности.

При решении проблемы экономного построения отдельных блоков ЭВМ естественными мерами сложности являются число элементов в схеме, число букв в формуле, число контактов в контактной схеме, число коньюнкций в дизьюнктивной нормальной форме и т.п.

При теоретическом изучении быстродействия программ естественной мерой сложности выступает *время вычисления* — число элементарных шагов, выполняемых алгоритмом, например машиной Тьюринга. Другими мерами сложности являются *длина программы* (например, число команд в ней), *объем рабочей памяти* — число ячеек памяти, используемой программой, время при параллельных вычислениях и т.п.

Имеется огромное разнообразие вычислительных задач и разработано много различных алгоритмов для них. Существующие языки программирования удобны при разработке и реализации алгоритмов, но мало пригодны для сравнительного анализа сложности алгоритмов. Поэтому теория сложности предпочитает иметь дело *с единым представлением задач и с простейшими моделями алгоритмов*, облегчающими их анализ. При этом важным является удачный выбор

- единого подхода представления задач и
- единой модели описания алгоритмов.

1.4 Контактные схемы

Контактные схемы — это наиболее “древний” класс управляющих систем, которые рассмотрены еще в первой работе Шеннона. Мы приводим определение контактной схемы, следуя

книге [8]. Пусть дан граф, в котором отмечены две вершины a и b (полюсы графа), а каждому ребру приписана одна буква алфавита $x_1, \ldots, x_n, \overline{x}_1, \ldots, , \overline{x}_n$. В обозначениях конъюнкций принято(см., например, книгу [13]) применять функцию x^σ, которая определяется следующим образом:

$$x^\sigma = \begin{cases} x, & \sigma = 1 \\ \overline{x}, & \sigma = 0 \end{cases}$$

Ребро с приписанной буквой $X_i^{\sigma_i}$ называется *контактом* переменной x_i, а именно *замыкающим*, если $\sigma_i = 1$, и *размыкающим*, если $\sigma_i = 0$. Описанный граф называется *контактной схемой* и обозначается. В контактной схеме допускаются "параллельные" контакты, т.е. контакты с двумя общими концами. Вершины, отличные от полюсов контактной схемы, называются внутренними.

Представление булевой функции в контактных схемах Говорят, что контакт x_i замкнут (проводит) при $x_i = 1$ и разомкнут при $x_i = 0$. Размыкающий контакт $\overline{x}_i$ ведет себя противоположным образом. Тогда цепь, состоящая из контактов $x_{i_1}^{\sigma_1}, \ldots, x_{i_k}^{\sigma_k}$, проводит тогда и только тогда, когда коньюнкция $x_{i_1}^{\sigma_1}, \ldots, x_{i_k}^{\sigma_k}$ равна 1. Это означает, что функция, реализуемая контактной схемой, есть проводимость схемы: функция равна 1 тогда и только тогда, когда есть проводящая цепь между полюсами.

1.5 Детерминированная ветвящаяся программа

Детерминированная ветвящаяся программа DBP (Deterministic Branching Program) с одной стороны — частный случай контактных схем, с другой стороны, их можно рассматривать как машинную модель вычислений. Мы приводим определение ветвящейся программы следуя книге [65].

Определение 1.5.1. DBP над множеством переменных $X = \{x_1, \ldots, x_n\}$ — это ориентированный ациклический граф, вершины которого делятся на множество внутренних и множество финальных вершин. Финальные вершины не имеют исходящих ребер и помечены нулем или единицей соответственно. Каждой внутренней вершине соответствует переменная $x \in X$, каждая внутренняя вершина имеет два исходящих ребра, помеченные 0 ($x = 0$) и 1 ($x = 1$), соответственно.

Такое задание DBP будем называть "графовым" заданием DBP.

Представление булевой функции. DBP представляет (вычисляет) булеву функцию $f : \{0,1\}^n \to \{0,1\}$ следующим образом. Вычисление значения $f(\sigma)$ для входного набора $\sigma \in \{0,1\}^n$ начинается из выделенной начальной вершины. Для каждой внутренней вершины, помеченной переменной x_j, осуществляется переход из этой вершины либо по 0-ребру, либо по 1-ребру, в соответствии со значением σ_j, которое принимает переменная x_j во входном наборе, до тех пор, пока не будет достигнута конечная вершина. Значение функции $f(\sigma)$ для входа σ — это значение достигнутой конечной вершины.

- *Сложность* $S(P)$ программы P — это число ее внутренних вершин.

- *Сложность* $S(f)$ реализации функции f в ветвящихся программах определяется как минимум сложности по всем ветвящимся программам P вычисляющих f:

$$S(f) = \min S(P).$$

Обозначим **P-BP** множество булевых функций, вычислимых ветвящимися программами полиномиальной (от числа переменных функции) сложности. Известно, что

$$\mathbf{P\text{-}BP} = \mathbf{L/poly},$$

где $\mathbf{L/poly}$ — это неоднородный класс сложности, содержащий задачи, решаемые детерминированными машинами Тьюринга с логарифмической памятью, имеющими "полиномиальную подсказку" (вспомогательную ленту с заранее записанной информацией полиномиального объема).

Нижние оценки сложности Известно, что почти все булевы функции экспоненциально сложны, т.е. почти все булевы функции нельзя представить программами сложности меньше чем $O\left(\frac{2^{n-1}}{n}\right)$.

Наиболее высокая нижняя оценка сложности для явно заданной булевой функции была получена Э. И. Нечипоруком в 1966 году. Для ветвящихся программ эту оценку впервые переложил П. Пудлак [53] в 1984 году.

Подфункция. Пусть $S \subseteq X_n$ – подмножество множества входных переменных. Все подфункции функции $f \in B_n$, полученные заменой всех переменных из $Z = X_n \backslash S$ на некоторые константы называются подфункциями f на множестве S.

Идея метода Нечипорука состоит в том, что функция, имеющая много разных подфункций, не может быть представлена ветвящейся программой "маленького размера".

Определение 1.5.2. Пусть $S \subseteq X_n$ - подмножество множества входных переменных. Все подфункции функции $f \in B_n$, полученные заменой всех переменных из $X_n \backslash S$ на некоторые константы называются подфункциями f на множестве S.

Теорема 1.5.1. **(Нечипорука)**
Пусть функция $f \in \mathbb{B}_n$ зависит от всех аргументов существенным образом. Пусть $S_1, ...S_k \subseteq X_n$ - попарно непересекающиеся множества входных переменных. Пусть s_i - количество подфункций f на S_i. Тогда

$$S(f) = \Omega\left(\sum_{i=1}^{k} \frac{\log s_i}{\log\log s_i}\right).$$

Максимально высокая нижняя оценка, получаемая применением теоремы Нечипорука, для индивидуальной функции (функции ISA [65]) от n переменных есть

$$O\left(\frac{n^2}{\log^2 n}\right).$$

Забывающие ветвящиеся программы Ветвящаяся программа называется *уровневой*, если ее вершины могут быть разбиты на уровни $0, 1, \ldots$ таким образом, что для каждого i ребра из вершин уровня i ведут только в вершины уровня $(i+1)$.

Ширина $w(P)$ уровневой ветвящейся программы P – это максимум от количества вершин на уровне, взятый по всем уровням программы P.

Глубина (длина) $l(P)$ уровневой ветвящейся программы P – это число уровней программы P.

Уровневая ветвящаяся программа P называется *забывающей*, если на любом уровне P тестируется только одна переменная. Каждая ветвящаяся программа P может быть преобразована (с полиномиальным усложнением) в забывающую ветвящуюся программу P', вычисляющую ту же самую функцию. Далее в данной работе мы будем рассматривать забывающие ветвящиеся программы.

1.6 Вероятностная ветвящаяся программа

Вероятностная ветвящаяся программа PBP (Probabilistic Branching Program) – это ветвящаяся программа, в которой каждая внутренняя вершина имеет выходную степень ≥ 2. При этом из каждой внутренней вершины выходит два типа ребер – помеченные 0 и 1. Каждому ребру e приписана вероятность $p(e)$ $(0 \leq p(e) \leq 1)$. Для каждой вершины сумма вероятностей всех ребер, исходящих из этой вершины, помеченных 0 (1), равна 1. Вычисление на входном наборе $\sigma \in \{0,1\}^n$ осуществляется следующим образом. На каждом шаге, начиная с выделенной начальной вершины, PBP считывает значение переменной, приписанной вершине, и в зависимости от значения считанной переменной переходит в следующие вершины либо по 0-ребрам, либо по 1-ребрам с вероятностями, приписанными соответствующим ребрам. Вероятность $p_{acc}(\sigma)$ принятия PBP входа σ – это вероятность того, что вычисление на входе σ приведет в конечную принимающую вершину (вершину, помеченную 1).

Определение 1.6.1. Пусть $\varepsilon > 0$. PBP P вычисляет функцию f с надежностью $(1/2+\varepsilon)$, если для $\sigma \in f^{-1}(1)$ $p_{acc}(\sigma) \geq 1/2+\varepsilon$, и для $\sigma' \in f^{-1}(0)$ $p_{acc}(\sigma') \leq 1/2-\varepsilon$. При этом говорят, что *P вычисляет функцию f с ограниченной ошибкой.*

Известно, что для ветвящихся программ общего вида (без ограничения на число считываний переменных) вероятностные модели не имеют существенных преимуществ в сложности реализации функций. А именно, по произвольной PBP, вычисляющей функцию с ограниченной ошибкой можно построить детерминированную ветвящуюся программу (с полиномиальным увеличением сложности), вычисляющую ту же функцию. Отметим, что в основе построения DBP по исходной PBP существенно используется прием полиномиального "повторения" PBP, что приводит к новой PBP, вычисляющей ту же булеву функцию с "очень небольшой" ошибкой.

Обозначим **BPP-BP** множество булевых функций, вычислимых PBP с ограниченной ошибкой и полиномиальной сложности. Таким образом справедливо равенство

$$\textbf{BPP-BP} = \textbf{P-BP}$$

Это равенство нарушается в моделях ветвящихся программ с ограничениями на число считываний переменных.

1.7 Ветвящиеся программы с ограничениями

Ветвящиеся программы – удобная модель для определения различных ограниченных вариантов. Целый ряд таких ограниченных моделей широко используются на практике.

1.7.1 Читающие один раз ветвящиеся программы

Ветвящаяся программа P называется читающей один раз (read-once), если на каждом пути каждая переменная $x \in X$ тестируется не более одного раза.

Для читающих один раз программ в 80-х годах были различными авторами доказаны экспоненциальные нижние оценки сложности реализации индивидуальных функций [65].

1.7.2 Читающие k раз ветвящиеся программы

Ветвящаяся программа P называется читающей k раз (read-k), если на каждом пути каждая переменная $x \in X$ тестируется не более k раз.

Для читающих k раз программ в были доказаны экспоненциальные нижние оценки сложности реализации индивидуальных функций [65].

1.7.3 Упорядоченные ветвящиеся диаграммы решений (OBDD)

Упорядоченная ветвящаяся диаграмма решений (OBDD) — это забывающая, читающая один раз ветвящаяся программа.

Таким образом, OBDD — это частный случай ветвящейся программы, все пути которой имеют одинаковую длину. Число вершин OBDD P на уровне i будем обозначать $width_i(P)$ и называть шириной уроня i. Шириной OBDD P — обозначение $width(P)$ — будем называть величину

$$width(P) = \max_i width_i(P).$$

Ширину $width(P)$ OBDD P будем считать сложностью OBDD. Понятно, что сложность OBDD P в смысле числа общего числа вершин OBDD P равна $width(P) \cdot n$, где n — это число переменных OBDD.

Существует естественная связь между моделью OBDD и моделью конечных автоматов. В ряде случаев в англоязычной литературе OBDD, называют неоднородными автоматами.

1.7.4 k-OBDD

Под k-OBDD понимается забывающая, читающая k раз ветвящаяся программа.

Таким образом, k-OBDD можно разделить на k слоев, каждый из которых будет представлять собой OBDD, причем порядок считывания один и тот же.

1.7.5 Практическая значимость OBDD

Развитие технологий в области конструирования и тестирования интегральных схем привело к развитию теории ветвящихся программ с ограничениями на число считываний переменных. Специальной такой ветвящейся программой является упорядоченная бинарная диаграмма решений – OBDD (Ordered Binary Decision Diagram). Она широко используется

для тестирование правильности проектирования СБИС [35], [64], программного и аппаратного оборудования, тестировании различных моделей и в других приложениях. (см. книгу [65]). Ключевым моментом такого применения OBDD является возможность представления функций в OBDD небольшой (полиномиальной) сложности

1.7.6 Линейная ветвящаяся программа

Определяемая в данном разделе линейная ветвящаяся программа [22] является обобщением понятия забывающей детерминированной программы и определяемой далее квантовой ветвящейся программы.

Линейная ветвящаяся программа LBP (Linear Branching Program) $\mathcal{P}$ над множеством переменных $X = \{x_1, \ldots, x_n\}$ и над d-мерным векторным пространством $\mathbf{V}^d$ есть тройка

$$\mathcal{P} = \langle T, |\mu_0\rangle, F\rangle \ .$$

- Множество $B^d = \{|1\rangle = (1, 0, \ldots, 0), \ \ldots, \ |d\rangle = (0, \ldots, 0, 1)\}$ базисных векторов будем называть базисными состояниями LBP.
- Вектора $|\mu\rangle = \sum_{i=1}^{d} z_i |i\rangle$ пространства $\mathbf{V}^d$ будем называть состояниями.
- Преобразования состояний $\mathcal{P}$ определяются последовательностью $T = (T_1, \ldots, T_\ell)$ (длины ℓ) инструкций. Каждая инструкция T_i это тройка $T_i = \{j_i, M_{j_i}(0), M_{j_i}(1)\}$, где j_i определяет переменную x_{j_i}, считываемую на шаге i, $M_{j_i}(0)$ и $M_{j_i}(1)$ — это $d \times d$ матрицы — линейные преобразования векторного пространства $\mathbf{V}^d$;
- $|\mu_0\rangle$ — начальное состояние программы $\mathcal{P}$;
- $F \subseteq B^d$ — подмножество базисных состояний, элементы которого будем называть принимающими состояниями. Элементы множества $\bar{F} = B^d \backslash F$ будем называть отвергающими состояниями. Через *Accept* и *Reject* будем обозначать множества индексов принимающих и отвергающих состояний соответственно: $Accept = \{i : |i\rangle \in F\}$ и $Reject = \{i : |i\rangle \in \bar{F}\}$.

Вычисление программы $\mathcal{P}$ на входе $\sigma = \sigma_1 \ldots \sigma_n \in \{0,1\}^n$ определяется следующим образом:

1. Вычисление $\mathcal{P}$ начинается из начального состояния $|\mu_0\rangle$;
2. На i-ом шаге вычисления $\mathcal{P}$ применяется инструкция T_i: если $x_{j_i} = \sigma_{j_i}$, то к текущему состоянию μ применяется преобразование $M_{j_i}(\sigma_{j_i})$ и программа $\mathcal{P}$ переходит в состояние $|\mu'\rangle = M_{j_i}(\sigma_{j_i}) |\mu\rangle$ (состояния программы представляются в виде вектор-столбцов);
3. Финальным состоянием (состоянием после последнего шага ℓ) будет состояние

$$|\mu(\sigma)\rangle = \prod_{i=\ell}^{1} M_{j_i}(\sigma_{j_i}) |\mu_0\rangle \ .$$

Определенная выше линейная программа является забывающей и имеет $\ell + 1$ уровень. Нумерация уровней начинается с нуля, последний (финальный) уровень имеет номер ℓ.

Шириной $w(\mathcal{P})$ LBP $\mathcal{P}$ будем называть размерность d пространства $\mathbf{V}^d$ состояний $\mathcal{P}$, а число ℓ называть длиной $l(\mathcal{P})$ программы $\mathcal{P}$.

Теперь забывающую DBP можно определить следующим образом.

1.7.7 Линейное представление забывающей DBP.

Забывающая DBP это LBP над векторным пространством $\mathbb{F}^d$, где $\mathbb{F}$ подходящее конечное поле. Множеством состояний такой LBP является множество B^d базисных состояний. Матрицы преобразований M задают преобразования множества B^d.

Входной набор σ принимается, если $|\mu(\sigma)\rangle \in F$.

1.8 Квантовая ветвящаяся программа

Квантовая ветвящаяся программа QBP (Quantum Branching Program) это LBP над комплекснозначным Гильбертовым d-мерным пространством $\mathcal{H}^d$. Состояниями QBP $\mathcal{Q}$ являются вектора $|\psi\rangle = \sum_{i=1}^{d} z_i |i\rangle$ с единичной нормой $|| \, |\psi\rangle \, || = \sqrt{\sum_{i=1}^{d} |z_i|^2} = 1$. Они называются чистыми состояниями или просто состояниями QBP. Преобразования QBP задаются комплекснозначными унитарными $d \times d$ матрицами.

Если $|\psi(\sigma)\rangle = \sum_{i=1}^{d} z_i |i\rangle$ — финальное состояние QBP после считывания входа σ, то вероятность $p_{acc}(\sigma)$ принятия входа σ программой P определяется как

$$p_{acc}(\sigma) = \sum_{i \in Accept} |z_i|^2.$$

Шириной квантовой ветвящейся программы Q называется размерность d пространства $\mathcal{H}^d$, а длиной – число l инструкций в последовательности T.

Определение 1.8.1. Пусть $\epsilon > 0$. Говорят, что QBP $\mathcal{Q}$ вычисляет функцию f с надежностью $1/2 + \epsilon$, если для $\sigma \in f^{-1}(1)$ выполняется неравенство $p_{acc}(\sigma) \geq 1/2 + \epsilon$, а для $\sigma' \in f^{-1}(0)$ — неравенство $p_{acc}(\sigma') \leq 1/2 - \epsilon$.

В частности, говорят, что квантовая ветвящаяся программа Q *вычисляет булеву функцию f с односторонней ошибкой*, если существует $\epsilon \in (0, 1)$ такое, что для любого $\sigma \in f^{-1}(1)$ вероятность принятия этого набора программой Q равна 1, а для любого $\sigma \in f^{-1}(0)$ вероятность принятия не превышает ϵ.

Заметим, что эта модель с одним измерением (“measure-once”) аналогична модели конечных квантовых автоматов, определенной в работе [49], в которой состояние системы изменяется унитарным образом за исключением финального измерения состояния.

Квантовые один раз читающие ветвящиеся программы.

Определение 1.8.2. Квантовая ветвящаяся программа Q называется QOBDD или один раз читающей квантовой ветвящейся программой, если каждая переменная $x \in \{x_1, \ldots, x_n\}$ появляется в последовательности инструкций T программы Q не более одного раза.

Обозначим через **P-QOBDD** класс функций, представимых квантовыми OBDD полиномиальной ширины.

Квантовые k раз читающие ветвящиеся программы.

Определение 1.8.3. Квантовая ветвящаяся программа Q называется k-QOBDD или k раз читающей QOBDD, если ее можно разбить на k последовательных подпрограмм, каждая из которых будет QOBDD. Причем для каждой подпрограммы порядок считывания переменных один и тот же.

Ветвящиеся программы ограниченной ширины. Определим классы сложности для ветвящихся программ ограниченной ширины.

- $\mathbf{P\text{-}BP}_k$ класс функций, представимых ветвящимися программами ширины k и полиномиальной длины.

  Положим $\mathbf{BWP\text{-}BP} = \cup_k \mathbf{P\text{-}BP}_k$.

- $\mathbf{BPP\text{-}BP}_k$ – подкласс класса **BPP-BP**, который состоит из функций, представимых вероятностными ветвящимися программами полиномиального сложности и ширины k с ограниченной ошибкой.

  Обозначим $\mathbf{BWBPP\text{-}BP} = \cup_k \mathbf{BPP\text{-}BP}_k$.

- **BQP-BP** – класс функций, которые ϵ-принимаются квантовыми ветвящимися программами *полиномиальной сложности*, для некоторой константы $\epsilon \in (0, 1/2)$.

- $\mathbf{BQP\text{-}BP}_k$ – подкласс класса **BQP-BP**, который состоит из всех функций, которые ϵ-принимаются квантовыми программами *полиномиальной сложности* и ширины k, для некоторой константы $\epsilon \in (0, 1/2)$.

  Мы обозначаем $\mathbf{BWBQP\text{-}BP} = \cup_k \mathbf{BQP\text{-}BP}_k$.

- **EQP-BP** – класс всех функций, которые вычислимы некоторыми квантовыми ветвящимися программами *полиномиальной сложности без ошибки*, т. е. *с нулевой ошибкой.*

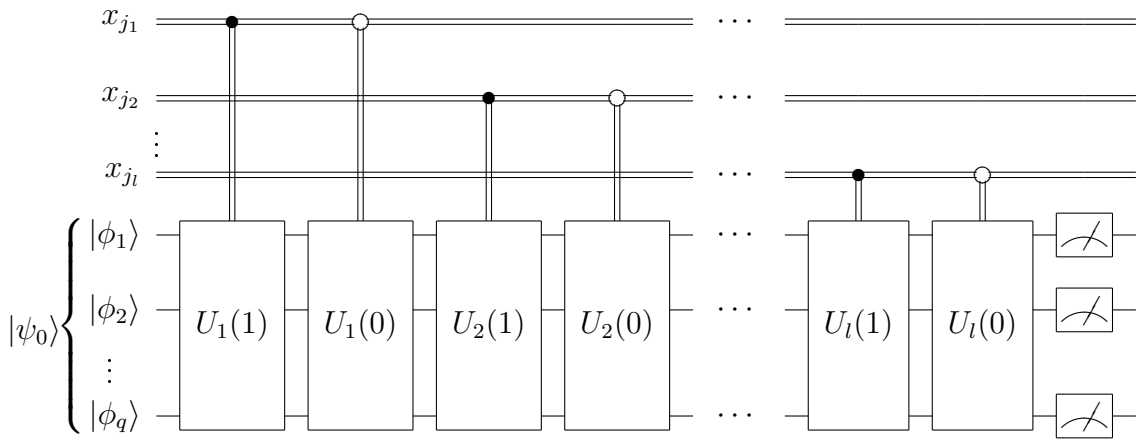
- $\mathbf{EQP\text{-}BP}_k$ – подкласс класса **EQP-BP**, который состоит из всех функций из всех функций, которые вычисляются квантовыми ветвящимися программами полиномиальной сложности и ширины k без ошибки.

  Обозначим $\mathbf{BWEQP\text{-}BP} = \cup_k \mathbf{EQP\text{-}BP}_k$.

1.8.1 Схемное представление

Предлагаемые нами методы построения эффективных квантовых алгоритмов в модели квантовых ветвящихся программ опираются на их схемное представление. А именно, квантовая ветвящаяся программа рассматривается как квантовая схема, дополненная возможностью считывать классические биты в качестве контролирующих для унитарных операторов. Таким

образом, любая квантовая схема есть квантовая ветвящаяся программа, которая существенно не зависит от своих классических входов.



Здесь $x_{j_1}, \ldots, x_{j_l}$ – последовательность переменных (необязательно различных), обозначающих классические управляющие сигналы. Согласно принятой в литературе по квантовым схемам нотации, классическое управление обозначается на схеме двойными проводами, а квантовое – одиночными.

Заметим, что для квантовой ветвящейся программы в схемном представлении явным образом проявляется еще одна мера сложности – число кубит q, необходимое для физической реализации соответствующей квантовой системы с классическим управлением. Согласно постулатам квантовой механики для реализации квантовой ветвящейся программы ширины d (системы с d состояниями) потребуется как минимум $\log d$ кубит.

Определение 1.8.4. Назовем квантовую ветвящуюся программу q-кубитной, если она может быть реализована в виде классически управляемой квантовой системы, основанной на q кубитах.

Замечание 1. Связь схем и квантовых ветвящихся программ также состоит в следующем. Согласно предложенному нами схемному представлению, квантовые ветвящиеся программы могут рассматриваться как квантовые схемы с возможностью считывать классические биты в качестве контролирующих. Поэтому любая квантовая схема может быть промоделирована квантовой ветвящейся программой с вырожденными инструкциями вида $\langle x_{j_i}, U_i, U_i \rangle$, т.е. при любом значении классической переменной будет применяться одно и то же унитарное преобразование.

1.8.2 Эффективные квантовые ветвящиеся программы.

По определению квантовых ветвящихся программ унитарные операторы, используемые в инструкциях, могут быть произвольными, в том числе решающими NP-трудные задачи. Поэтому разумным ограничением является полиномиальная по времени конструируемость этих преобразований, т.е. они должны быть представимы в виде произведения не более чем полиномиального числа "элементарных" унитарных операторов из некоторого конечного базиса (например, из "стандартного" базиса $\{H, S, \mathrm{CNOT}, \pi/8\}$, см. [9]).

В силу вышесказанного эффективными будем называть те квантовые ветвящиеся программы, ширина и длина которых не превосходит $n^{O(1)}$. Т.к. при ограничении ширины $n^{O(1)}$ число кубит, задействованных в вычислениях, составляет $q = O(\log n)$, каждое унитарное преобразование такой программы можно представить в виде произведения $O(q^2 4^q) = n^{O(1)}$

(т.е. полиномиального числа) "элементарных" операторов. Поэтому полиномиальное ограничение на ширину и длину программы влечет и полиномиальную сложность используемых унитарных преобразований.

Аналогично классическому случаю, известно, что для почти всех булевых функций сложность квантовых ветвящихся программ экспоненциальна. В последнее десятилетие рядом авторов получены результаты, показывающие, что для целого ряда функций квантовые ветвящиеся программы могут быть более эффективными, чем классические детерминированные и вероятностные ветвящиеся программы.

В данной работе мы ограничиваемся рассмотрением эффективных квантовых один раз читающих ветвящихся программ.

Заметим, что из этой оценки не следует превосходство квантовых OBDD над классическими.

В 1988 Баррингтон [4] показал, что $5-\mathbf{P\text{-}BP} = \mathbf{NC^1}$. Более того, для детерминированных ветвящихся программ ширина 5 необходима, если $\mathbf{NC^1} = \mathbf{ACC^0}$ [30],

где класс $\mathbf{ACC^0}$ – это класс вычислительных проблем, вычислимых однородными семействами булевских схем с дополнительными MOD m вентилями для произвольных m. Причем схемы эти имеют полиномиальный размер, константную глубину и неограниченное количество входов [[14]].

Ф.М. Аблаев, К. Мур и К. Полетт показали, что $\mathbf{EQP\text{-}BP}_2 = \mathbf{NC^1}$. Этот факт остается одним из наиболее интересных результатов в области квантовых ветвящихся программ по сей день. Как видно из приведенной ниже теоремы, все классы сложности, определенные выше, равны между собой.

Теорема 1.8.1 ([22]).

$$\mathbf{EQP\text{-}BP}_2 = \mathbf{BQP\text{-}BP}_2 = \mathbf{EQP\text{-}BP} = \mathbf{BQP\text{-}BP} = \mathbf{BPP\text{-}BP} = \mathbf{BWP\text{-}BP} = \mathbf{NC^1}.$$

Ветвящиеся программы без ограничений в общем слабо исследованы в силу отсутствия хороших методов доказательства для них. Ограничение ширины ветвящейся программы привело к открытию некоторых нетривиальных фактов о ветвящихся программах. Однако, ограничение ширины не привело к интересной иерархии классов сложности.

Оказывается, что для OBDD ситуация несколько более интересная. Определим классы сложности, с которыми мы будем иметь дело.

- **BQP-OBDD** – класс всех функций, вычислимых с ограниченной ошибкой квантовыми OBDD *полиномиальной сложности.*
- **EQP-OBDD** – класс всех функций, которые вычисляются некоторыми квантовыми OBDD *полиномиальной сложности без ошибки*, то есть с нулевой ошибкой.
- **RQP-OBDD** – класс всех функций, которые вычисляются с односторонней ошибкой некоторыми квантовыми OBDD *полиномиальной сложности.*
- Класс квантовых OBDD с *нулевой ошибкой* опреляется так

$$\mathbf{ZQP\text{-}OBDD} = \mathbf{RQP\text{-}OBDD} \cap \mathbf{coRQP\text{-}OBDD}.$$

- Класс всех *обратимых* программ из **P-OBDD** называется **Rev-OBDD**.

Последний *детерминированный* класс сложности, определенный выше, является естественным детерминированным подклассом **BQP-OBDD**, точно также, как **P-OBDD** – это естественный подкласс **BPP-OBDD**.

Следующий результат был получен М. Зауерхоффом и Д. Зиилингом [56].

Теорема 1.8.2 ([56]). **Rev-OBDD=EQP-OBDD=ZQP-OBDD**.

Однако, оставшиеся классы сложности для OBDD не схлопываются аналогично с соответствующими классами для ветвящихся программ ограниченной ширины.

В 2001 Ф.М. Аблаев, А.Ф. Гайнутдинова и М. Карпински [15] предложили явно заданную функцию, для которой между *стабильными вероятностными* и *стабильными квантовыми* OBDD обнаружилась экспоненциальная разница в сложности. *Стабильность* довольно сильное ограничение для OBDD.

Определение 1.8.5. Рассмотрим OBDD $\mathcal{P}$, где $T = (j_i, M_i(0), M_i(1))_{i=1}^n$ – подходящая последовательность преобразований и n – это длина входа. Тогда $\mathcal{P}$ называется *стабильной*, если $M_i(0) = M_j(0)$ и $M_i(1) = M_j(1)$ для всех $i, j \in \{1, \ldots, n\}$. Другими словами, преобразования не зависят от уровня $\mathcal{P}$.

Определение 1.8.6 ([15]). Функция MOD_p определена следующим образом. На входе $\sigma = \sigma_1, \ldots, \sigma_n \in \{0,1\}^n$ $\mathrm{MOD}_p(\sigma) = 1$, тогда и только тогда когда число единиц в σ делится на p.

Теорема 1.8.3 ([15]). Функция MOD_p представима *стабильной*, один раз читающей, ширины-$O(\log p)$ квантовой ветвящейся программой с односторонней ошибкой $\epsilon > 0$.

Любая *стабильная* вероятностная OBDD, вычисляющаяMOD_p имеет ширину по крайней мере p.

С этим результатом родилась надежда, что удастся строго доказать превосходство вычислительной мощи *квантовых компьютеров* над *вероятностными* для OBDD, которые являются широко используемой практически значимой моделью вычислений. Однако, в 2004 году М. Зауэрхофф и Д. Зиилинг показали, что *квантовые* и *классические* OBDD несравнимы [56]. В доказательстве данного результата они использовали следующие функции.

Определение 1.8.7. Функция *"Проверка перестановочности матрицы"* функция PERM_n определена на булевских матрицах размера $n \times n$, строки которых выстроены в одну строку друг за другом. На входе $\sigma \in \{0,1\}^{n^2}$ $\mathrm{PERM}_n(\sigma) = 1$, тогда и только тогда когда σ соответствует перестановочной матрице. То есть, матрице из единиц и нулей, в которой каждая строка и каждый столбец содержит ровно одну единицу.

Известно, что неограниченные модели *недетерминированных* один раз читающих программ для функции PERM_n имеют экспоненциальный размер (см. [47, 43]). Однако, известен следующий результат [55, 65].

Теорема 1.8.4 ([55]). Функция PERM_n может быть вычислена с односторонней ошибкой $\epsilon(n)$ вероятностной один раз читающей упорядоченной ветвящейся программой размера

$$O\left(\epsilon(n)^{-2} n^5 \log^3 n\right).$$

Кроме того, М. Зауэрхофф и Д. Зиилинг определили элегантную функцию *соседствующих единиц*.

Определение 1.8.8 ([56]). Функция *соседствующих единиц* NO_n определена на булевских переменных $x_1, \ldots, x_n$. Она принимает значение 1, тогда и только тогда, когда имеются две соседствующие переменные, значения которых на входе равны 1. То есть, существует индекс $i \in \{1, \ldots, n-1\}$, такой что $x_i = x_{i+1} = 1$.

Функция NO_n вычислима детерминированной OBDD размера $O(n)$. Несравнимость *детерминированных* и *квантовых* OBDD показывают следующие результаты.

Теорема 1.8.5 ([56]). Существует *квантовая* OBDD, вычисляющая $\neg\mathrm{PERM}_n$ с односторонней ошибкой $1/n$. Причем, размер такой программы будет $O(n^6 \log n)$. Таким образом, следующее верно

$$\textbf{BQP-OBDD} \not\subseteq \textbf{P-OBDD}.$$

Теорема 1.8.6 ([56]). Размер любой *квантовой* OBDD G, вычисляющей NO_n с ограниченной ошибкой не меньше $2^{\Omega(n)}$. Таким образом, верно следующее.

$$\textbf{P-OBDD} \not\subseteq \textbf{BQP-OBDD}.$$

Некоторые наиболее важные известные соотношения между классами сложности показаны на **Схеме 1.1**.

Заметим, что хотя

$$\textbf{BPP-OBDD} \not\subseteq \textbf{BQP-OBDD},$$

остается открытым вопрос, верно ли что

$$\textbf{BQP-OBDD} \subseteq \textbf{BPP-OBDD}.$$

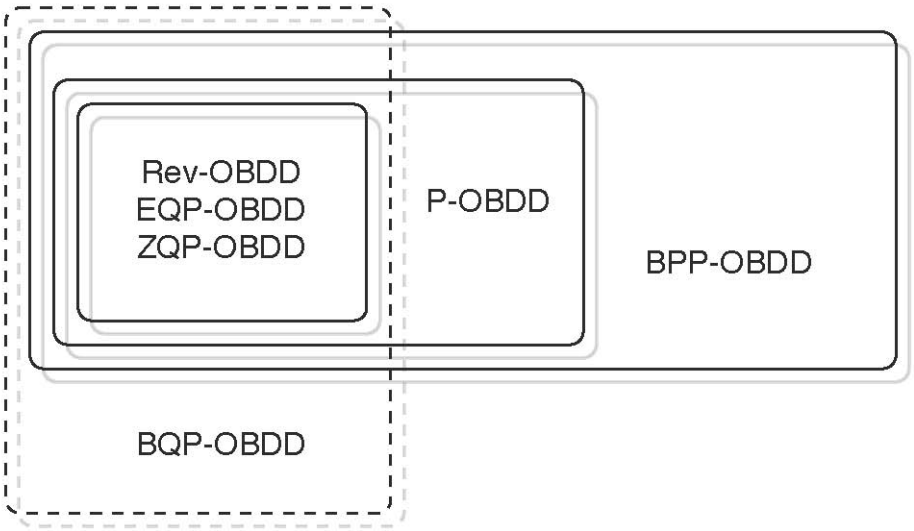


Рис. 1.1: Соотношения классов сложности для OBDD

М. Зауэрхоффом и Д. Зиилинг в [56] на примере функции косвенной адресации показали, что квантовые один раз читающие ветвящиеся программы могут быть мощнее квантовых OBDD.

Определение функции *косвенной адресации* практически дает нам алгоритм построения *дерева решений* размера $O(2^k \cdot b) = O(n^2 \log_2 n)$. Это дерево решений можно рассматривать и как один раз читающую *квантовую ветвящуюся программу*, так как в нем происходит *обратимое* вычисление. Однако, любая квантовая ZQP-OBDD, которая вычисляет ISA_n должна быть размера хотя бы $2^{\Omega(n/\log_2 n)}$ [56].

Недавно М. Зауэрхофф показал несравнимость также и для *незабывающих* ветвящихся программ [57]. В этой работе Зауэрхофф использовал хорошо известную функцию *непересечение множеств* и одну функцию, которую он придумал специально для этой цели.

Определение 1.8.9. Пусть $x = (y_1, \ldots, y_n)$, $y = (y_1, \ldots, y_n) \in \{0,1\}^n$. Определим функцию *непересечение множеств*

$$\mathrm{DISJ}_n(x,y) = \neg(x_1y_1 \vee \cdots \vee x_ny_n).$$

Определение 1.8.10 ([57]). Для некоторого положительного n и $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, пусть $p(n)$ – наименьшее простое число, большее n и пусть $s_n(x) = (\Sigma_{i=1}^n i \cdot x_i) \mod p(n)$. Определим функцию *взвешенной суммы* как $\mathrm{WS}_n(x) = x_{s_n(x)}$, если $s_n(x) \in \{1, \ldots, n\}$ и положим ее значение равным 0 в противном случае.

Для оставшейся части входа $y = (y_1, \ldots, y_n) \in \{0,1\}^n$ определим *смешанную взвешенную сумму* как $\mathrm{MWS}_n(x,y) = x_i \oplus y_i$, если $i = s_n(x) = s_n(y) \in \{1, \ldots, n\}$ и положим ее значение равным 0 в противном случае.

Теперь сформулируем разделение классов сложности в двух теоремах.

Теорема 1.8.7 ([57]). Каждая *вероятностная* один раз читающая ветвящаяся программа, вычисляющаяMWS_n с двухсторонней ошибкой, ограниченной произвольной константой, меньшей $1/2$ будет иметь размер не меньше $2^{\Omega(n)}$, тогда как MWS_n может быть вычислена *квантовой* один раз читающей ветвящейся программой размера $O(n^3)$ с нулевой ошибкой.

Теорема 1.8.8 ([57]). Каждая квантовая один раз читающая ветвящаяся программа, вычисляющаяDISJ_n с двусторонней ошибкой, ограниченной константой, меньшей $1/2 - 2\sqrt{3}/7$ (≈ 0.005) будет иметь размер $2^{\Omega(n)}$

Непересечение множеств вычислимо детерминированной OBDD тривиальным образом для произвольного порядка тестирования.

Те несколько результатов разделения классов сложности, которые мы представили в этом разделе полностью разрушают надежду на то, что квантовые OBDD окажутся мощнее своих классических аналогов для всех вычислительных проблем. Однако, это не значит, что для каких-то конкретных практически значимых задач мы не найдем квантовый алгоритм, значительно превосходящий любые возможные классические аналоги в области вычислительной сложности.

Глава 2

Методы построения эффективных квантовых алгоритмов и нижние оценки сложности их представления

2.1 Обобщенная нижняя оценка

В данном разделе доказывается нижняя оценка сложности представления функций в квантовых ветвящихся программах специального вида — синтаксических КВП. Эта нижняя оценка показывает, что для произвольной булевой функции квантовая ветвящаяся программа, ее распознающая, не может быть "дешевле" логарифма сложности классической детерминированной программы для этой функции. Эта оценка используется ниже для доказательства оптимальности алгоритмов для квантовых OBDD.

На последующем этапе работы планируется описание достаточного условия для свойств функции, при котором справедлива "высокая" нижняя оценка сложности ее реализации в один раз читающих квантовых программах. Т.е. будет получена нижняя оценка, показывающая, что для ряда индивидуальных функций вычисляющие их квантовые программы так же, как и классические детерминированные, экспоненциально сложны.

2.1.1 Синтаксические квантовые ветвящиеся программы.

Из определения КВП, представляющей функцию с надежностью $1/2+\epsilon$ следует, что финальные состояния (состояния ℓ-ого уровня) программы, достижимые на входных наборах σ из $\{0,1\}^n$ разбиваются на два множества

$$\mathcal{A} = \{|\psi(\sigma)\rangle : p_{acc}(\sigma) \geq 1/2+\epsilon\} \quad \text{и} \quad \mathcal{R} = \{|\psi(\sigma)\rangle : p_{acc}(\sigma) \leq 1/2-\epsilon\}.$$

Из этого следует, что расстояние $\rho(\mathcal{A},\mathcal{R})$ между множествами $\mathcal{A}$ и $\mathcal{R}$ оценивается снизу некоторой константой $\theta(\epsilon)$. (Расстояние между множествами определяется стандартно на основе метрики $\rho(|\psi\rangle, |\psi'\rangle) = || \, |\psi\rangle - |\psi'\rangle \, ||$).

В этом случае будем говорить, что множества $\mathcal{A}$ и $\mathcal{R}$ *изолированы*. В случае "много раз" читающих ветвящихся программ могут возникать *фиктивные состояния* (состояния, не достижимые при вычислениях на входных наборах). При этом фиктивные финальные состояния могут "разрушить" свойство изолированности множеств $\mathcal{A}$ и $\mathcal{R}$.

Определение 2.1.1. Мы определяем синтаксические квантовые ветвящиеся программы (СКВП) как программы, множество всех финальных состояний (достижимых и фиктивных) которых разбивается на два изолированных множества.

Непосредственно из определения 2.1.1 следует, что все финальные состояния $|\psi\rangle = (z_1, \ldots, z_d)$ (достижимые и фиктивные) СКВП $\mathcal{Q}$ разбиваются на два множества

$$[\mathcal{A}] = \left\{ |\psi\rangle : \sum_{i \in Accept} |z_i|^2 \geq 1/2 + \epsilon \right\} \quad \text{и} \quad [\mathcal{R}] = \left\{ |\psi\rangle : \sum_{i \in Accept} |z_i|^2 \leq 1/2 - \epsilon \right\}.$$

Отметим, что квантовая модель "один раз читающей" ветвящейся программы (каждая переменная на каждом пути вычисления может читаться только один раз), вычисляющая функцию f с надежностью $1/2 + \epsilon$, является синтаксической.

Теорема 2.1.1. Пусть функция f вычислима СКВП $\mathcal{Q}$ с надежностью $1/2 + \epsilon$. Тогда f вычислима детерминированной ветвящейся программой (ДВП) P, для которой выполняется равенство $l(P) = l(\mathcal{Q}) = \ell$ и

$$w(P) \leq \left(1 + \frac{1}{\epsilon}\right)^{2w(\mathcal{Q})}.$$

Доказательство теоремы 2.1.1 приводится в следующем разделе.

Положим $w(f) = \min\{w(P)\}$, где минимум берется по всем ДВП P, вычисляющих функцию f. Непосредственно из теоремы 2.1.1 следует нижняя оценка на $w(\mathcal{Q})$.

Свойство 2.1.1. Если функция f вычислима СКВП $\mathcal{Q}$ с надежностью $1/2 + \epsilon$, тогда

$$w(\mathcal{Q}) \geq c(\epsilon) \log w(f).$$

Реализация функции MOD_m в один раз читающих моделях ДВП и СКВП (в этом случае СКВП есть в точности КВП) показывает, что нижняя оценка свойства 2.1.1 точна с точностью до мультипликативной константы. ($MOD_m(\sigma) = 1$ тогда и только тогда, когда число единиц в наборе σ кратно m).

- Легко проверяется [17], что для произвольной один раз читающей ДВП P, вычисляющей MOD_m, выполняется неравенство $w(P) \geq m$.
- С другой стороны, в [23] приводится один раз читающая КВП $\mathcal{Q}$, вычисляющая MOD_m с надежностью $1/2 + \epsilon$, для которой выполняется равенство $w(\mathcal{Q}) = O(\log m)$.

2.1.2 Доказательство теоремы 2.1.1

Доказательство теоремы состоит из двух этапов. На первом этапе по СКВП $\mathcal{Q}$ строится ДВП DP экспоненциальной (от ℓ) ширины, вычисляющая ту же функцию f. На втором этапе по ДВП DP строится искомая ДВП P.

Первый этап (построение ДВП DP).

Вычисление программы $\mathcal{Q}$ на наборах $\sigma \in \{0,1\}^n$ — это ℓ-шаговые линейные преобразования состояний, начинающиеся с начального состояния $|\psi_0\rangle$. Все возможные вычисления программы $\mathcal{Q}$ на входных наборах из $\{0,1\}^n$ представляются $(\ell+1)$-уровневой забывающей ДВП

DP, задаваемой в виде полного $(\ell+1)$-уровневого бинарного дерева. Вершины программы DP помечаются состояниями $|\psi\rangle$ программы $\mathcal{Q}$. Уровень 0 содержит начальную вершину DP, помеченную начальным состоянием $|\psi_0\rangle$ программы $\mathcal{Q}$. Уровень $i \in \{0,\ldots,\ell\}$ представляет из себя i-ый шаг вычисления. Из каждой вершины $|\psi\rangle$ уровня i, $i \in \{0,\ldots,\ell-1\}$, исходят два ребра помеченные $x_{j_i}=0$ и $x_{j_i}=1$, где x_{j_i} — переменная, считываемая на шаге i. Ребро $x_{j_i}=\gamma$ ведет из вершины $|\psi\rangle$ уровня i в вершину $|\psi'\rangle$ уровня $i+1$, если СКВП $\mathcal{Q}$, находясь на шаге i в состоянии $|\psi\rangle$, переходит в состояние $|\psi'\rangle$ при считывании $x_{j_i}=\gamma$.

Вершины ℓ-ого уровня являются финальными. Финальные вершины $|\psi\rangle \in [\mathcal{A}]$ дополнительно помечаются единицей (принимающие вершины программы DP), а вершины $|\psi\rangle \in [\mathcal{R}]$ дополнительно помечаются нулем (отвергающие вершины программы DP).

Непосредственно из описания программы DP следует

Свойство 2.1.2. ДВП DP вычисляет ту же функцию f, что и СКВП $\mathcal{Q}$ и имеет следующие характеристики: $l(DP)=\ell$ и $w(DP)=2^\ell$.

Метрические свойства ДВП DP.

Следующие понятия и факты теории метрических пространств приведены в книге [3]. Пусть $\mathcal{M}$ — это метрическое пространство с метрикой ρ. Говорят, что точки μ,μ' из $\mathcal{M}$ связаны θ-цепью, если существует конечное множество $\mu_1,\mu_2,\ldots,\mu_m$ точек из $\mathcal{M}$ таких, что $\mu_1=\mu, \mu_m=\mu'$ и $\rho(\mu_i,\mu_{i+1})<\theta$ для $i\in\{1,\ldots,m-1\}$. Подмножество $\mathcal{C}\subseteq\mathcal{M}$ называется θ-компонентой, если произвольные две точки $\mu,\mu'\in\mathcal{C}$ связаны θ-цепью.

Обозначим через Ψ_i, $i\in\{0,\ldots,\ell\}$, множество всех состояний (вершин) программы DP уровня i. На множестве Ψ_i определим метрику ρ по формуле $\rho(|\psi\rangle,|\psi'\rangle)=||\,|\psi\rangle-|\psi'\rangle\,||$. Для $\theta>0$ число θ-компонент множества Ψ_i, $i\in\{0,\ldots,l\}$, зависит от строения множества Ψ_i (может оказаться, что все множество Ψ_i представляет собой одну θ-компоненту). Следующее свойство дает верхнюю оценку числа возможных θ-компонент множества Ψ_i.

Свойство 2.1.3. Для $i\in\{0,\ldots,\ell\}$, $\theta>0$ число t_i θ-компонент множества Ψ_i оценивается сверху величиной

$$t_i \le \left(1+\frac{2}{\theta}\right)^{2w(\mathcal{Q})}.$$

Доказательство. Обозначим через $\mathcal{C}_i$ множество всех θ-компонент множества Ψ_i. В каждой θ-компоненте $C\in\mathcal{C}_i$ выберем одну точку $|\alpha\rangle\in C$. Если рассмотреть сферы радиуса $\theta/2$ с центрами в таких точках $|\alpha\rangle\in C$, тогда все эти сферы попарно не пересекаются и могут иметь общие точки лишь на границах. Все эти сферы находятся в большей сфере радиуса $1+\theta/2$ с центром в $|(0,0,\ldots,0)\rangle$. Объем сферы радиуса r в комплексном пространстве $\mathcal{H}^d$ is равен cr^{2d} (в комплексно-значном пространстве $\mathcal{H}^d$ каждая точка $|\alpha\rangle$ имеет размерность $2d$). Константа c зависит от используемой метрики пространства $\mathcal{H}^d$. Таким образом, имеем

$$t_i \le \frac{c\,(1+\theta/2)^{2d}}{c\,(\theta/2)^{2d}} = \left(1+\frac{2}{\theta}\right)^{2w(\mathcal{Q})}.$$

□

На каждом уровне $i\in\{0,\ldots,\ell-1\}$ преобразования состояний $|\psi\rangle$ программы DP задаются унитарной $(d\times d)$-матрицей, которая определяется значением γ считываемой переменной x_{j_i}. Для подмножества $D\subseteq\Psi_i$ и унитарной $(d\times d)$-матрицы M положим $D'=\{|\psi'\rangle : |\psi'\rangle = M\,|\psi\rangle, |\psi\rangle\in D\}$ (множество $D'=M(D)$ — образ D для преобразования M). Следующее

утверждение показывает, что программа DP, сохраняет свойство принадлежности состояний одной θ-компоненте при унитарных преобразованиях.

Свойство 2.1.4. Для ДВП DP, для $i \in \{0, \ldots, \ell - 1\}$, $\theta > 0$, для произвольной θ-компоненты C множества Ψ_i и произвольной унитарной $(d \times d)$-матрицы M множество $M(C)$ является подмножеством некоторой θ-компоненты C' множества Ψ_{i+1}.

Доказательство. Унитарное преобразование сохраняет расстояние ρ между векторами. Следовательно, если состояния $|\psi\rangle$ и $|\mu\rangle$ входят в одну θ-компоненту $C \in \mathcal{C}_i$, то их образы $|\psi'\rangle = M|\psi\rangle$ и $|\mu'\rangle = M|\mu\rangle$ входят в одну θ-компоненту $C' \in \mathcal{C}_{i+1}$. □

Следующее утверждение показывает, что для $\theta = 2\epsilon$ множество $[\mathcal{A}]$ принимающих вершин и множество $[\mathcal{R}]$ отвергающих вершин программы DP являются объединениями θ-компонент множества Ψ_ℓ состояний уровня ℓ.

Свойство 2.1.5. Пусть $\mathcal{C}_\ell = \{C_1, \ldots, C_t\}$ — это множество θ-компонент Ψ_ℓ для $\theta = 2\epsilon$. Тогда

$$[\mathcal{A}] = \bigcup_{i \in I} C_i \qquad \text{и} \qquad [\mathcal{R}] = \bigcup_{i \in J} C_i,$$

где $I \cup J = \{1, \ldots, t\}$ и $I \cap J = \emptyset$.

Доказательство. В силу определения ДВП DP имеем, что множество Ψ_ℓ состояний уровня ℓ разбивается на множество $[\mathcal{A}]$ принимающих вершин и множество $[\mathcal{R}]$ отвергающих вершин программы DP. Покажем, что для произвольной θ-компоненты $C \in \mathcal{C}_\ell$ выполняется одно из двух включений $C \subseteq [\mathcal{A}]$ или $C \subseteq [\mathcal{R}]$. Для этого достаточно показать отсутствие θ-цепей между точками множеств $[\mathcal{A}]$ и $[\mathcal{R}]$, т.е., что для произвольных $|\psi\rangle \in [\mathcal{A}]$ и $|\psi'\rangle \in [\mathcal{R}]$ справедливо соотношение

$$\rho(|\psi\rangle, |\psi'\rangle) \geq \theta = 2\epsilon. \tag{2.1}$$

Пусть $|\psi\rangle = (z_1, \ldots, z_d)$ и $|\psi'\rangle = (z'_1, \ldots, z'_d)$. Тогда имеем

$$2\epsilon \leq \sum_{i \in Accept} (|z_i|^2 - |z'_i|^2) = \sum_{i \in Accept} (|z_i| - |z'_i|)(|z_i| + |z'_i|) \leq \sum_{i \in Accept} (|z_i - z'_i|)(|z_i| + |z'_i|).$$

Аналогично предыдущему имеем

$$2\epsilon \leq \sum_{i \in Reject} (|z'_i|^2 - |z_i|^2) = \sum_{i \in Reject} (|z'_i| - |z_i|)(|z_i| + |z'_i|) \leq \sum_{i \in Reject} (|z_i - z'_i|)(|z_i| + |z'_i|).$$

Объединяя два этих неравенства, получаем

$$4\epsilon \leq \sum_{i=1}^{d} (|z_i - z'_i|)(|z_i| + |z'_i|). \tag{2.2}$$

Применяя неравенство Коши-Буняковского $\left(\sum_{i=1}^d a_i b_i \leq \sqrt{\sum_{i=1}^d a_i^2}\sqrt{\sum_{i=1}^d b_i^2}\right)$ из (2.2) получаем

$$4\epsilon \leq ||\,|\psi\rangle - |\psi'\rangle\,|| \sqrt{\sum_{i=1}^{d} (|z_i| + |z'_i|)^2}.$$

Из неравенства Коши-Буняковского также непосредственно следует, что

$$\sqrt{\sum_{i=1}^{d}(|z_i| + |z_i'|)^2} \leq || \, |\psi\rangle \, || + || \, |\psi'\rangle \, || = 2$$

Два последних неравенства доказывают (2.1). □

Второй этап (Построение ДВП P).

Построение искомой ДВП P основывается на свойствах предыдущего раздела. Положим $\theta = 2\epsilon$. Программа P это $(\ell+1)$-уровневая забывающая ветвящаяся программа. На уровне j, считывается переменная x_{i_j} (как в программе DP). Вершинам уровня j соответствуют θ-компоненты из $\mathcal{C}_j$. Из вершины $C \in \mathcal{C}_j$ ребро, помеченное $x_{i_j} = \gamma$, ведет в вершину $C' \in \mathcal{C}_{j+1}$, если $M_j(\gamma)(C) \subseteq C'$. Вершина $C \in \mathcal{C}_\ell$ последнего уровня ℓ дополнительно помечается единицей (нулем), если $C \subseteq [\mathcal{A}]$ ($C \subseteq [\mathcal{R}]$).

Из описания ДВП P следует, что P вычисляет ту же функцию f что и ДВП DP и ее ширина $w(P)$ оценивается сверху:

$$w(P) \leq \max_{0 \leq i \leq \ell} |\mathcal{C}_i| \leq \left(1 + \frac{1}{\epsilon}\right)^{2w(\mathcal{Q})}.$$

Теорема 2.1.1 полностью доказана.

Нижняя оценка Положим $w(f) = \min\{w(P)\}$, где минимум берется по всем ДВП P, вычисляющих функцию f. Непосредственно из теоремы 2.1.1 следует нижняя оценка на $w(\mathcal{Q})$.

Свойство 2.1.6. Если функция f вычислима СКВП $\mathcal{Q}$ с надежностью $1/2 + \epsilon$, тогда

$$w(\mathcal{Q}) \geq c(\epsilon) \log w(f).$$

2.2 Методы построения эффективных квантовых алгоритмов

2.2.1 Квантовый метод отпечатков

Метод отпечатков (Fingerprinting) – это техника, которая позволяет представлять объекты (слова в некотором конечном алфавите) их образами (*отпечатками, fingerprints*), значительно более компактными, чем оригиналы. Кроме того, она позволяет с высокой вероятностью извлекать информацию о входном наборе.

Предложенный нами вариант *метода отпечатков (fingerprinting)* предназначен для построения эффективных по памяти квантовых алгоритмов в квантовых моделях вычислений с классическим управлением, таких как *упорядоченные один раз читающие квантовые ветвящиеся программы (OBDD)* [2, 23].

Ключевым моментом использования предложенного метода является представление вычисляемых булевых функций *характеристическими полиномами.*

Определение 2.2.1. Назовем полином g_f над некоторым кольцом вычетов $\mathbb{Z}_m$ характеристическим для булевой функции $f(x_1, \ldots, x_n)$, если для любого $\sigma \in \{0,1\}^n$ выполняется: $f(\sigma) = 1 \iff g_f(\sigma) = 0$.

Такой полином существует для каждой булевой функции f – его можно построить по произвольной ДНФ для отрицания f следующими заменами:

$$x_i \to x_i$$

$$\overline{x}_i \to (1 - x_i)$$

$$\vee \to +$$

$$\& \to \cdot$$

Однако, это не единственный способ получить характеристический полином, и нам требуется выбрать тот, что обладает нужным свойством, а именно – линейностью.

Техника отпечатков. Для решаемой задачи фиксируется допустимая вероятность ошибки $\epsilon \in (0, 1)$ и выбирается характеристический полином g над кольцом $\mathbb{Z}_m$ (где m выбирается подходящим способом).

Далее, для произвольного двоичного набора $\sigma = \sigma_1 \dots \sigma_n$ порождается его отпечаток $|h_\sigma\rangle$, соединяющий в себе t однокубитных отпечатков $|h^i_\sigma\rangle$:

$$\begin{aligned} |h^i_\sigma\rangle &= \cos \tfrac{2\pi k_i g(\sigma)}{m} |0\rangle + \sin \tfrac{2\pi k_i g(\sigma)}{m} |1\rangle \\ |h_\sigma\rangle &= \tfrac{1}{\sqrt{t}} \sum_{i=1}^{t} |i\rangle |h^i_\sigma\rangle \end{aligned}$$

Другими словами, последний из $\log t + 1$ кубита в квантовом регистре одновременно поворачивается на t различных углов вокруг оси $\hat{y}$ сферы Блоха. Такой квантовый параллелизм достигается за счет использования контролируемых операторов $C_i(R_i)$, которые применяют вращение R_i к последнему кубиту, если первые $\log t$ кубит находились в состоянии $|i\rangle$, а в противном случае применяется тождественный оператор I.

Предлагаемая техника нацелена на достоверное распознавание равенства нулю значения $g(\sigma)$. Для этого параметры $k_i \in \{1, \dots, m-1\}$ для всех $i = \overline{1, t}$ выбираются специальным образом, исходя из следующего определения.

Определение 2.2.2. Множество параметров $K = \{k_1, \dots, k_t\}$ называется "хорошим" для целого числа $l \neq 0 \bmod m$, если для некоторого $\epsilon \in (0, 1)$

$$\frac{1}{t^2} \left(\sum_{i=1}^{t} \cos \frac{2\pi k_i l}{m} \right)^2 < \epsilon.$$

Левая часть неравенства соответствует квадрату амплитуды базисного состояния $|0\rangle^{\otimes \log t} |0\rangle$ после применения оператора $H^{\otimes \log t} \otimes I$ к отпечатку $|h_\sigma\rangle$. Неформально, такое множество гарантирует, что вероятность ошибки будет ограничена константой ϵ.

Существование "хорошего" множества параметров Следующая лемма доказывает существование "хорошего" множества и развивает идеи соответствующего утверждения из [28].

Лемма 2.2.1. Существует множество K, где $|K| = t = 2^{\lceil \log((2/\epsilon) \ln 2m) \rceil}$, которое является "хорошим" для всех целых $l \neq 0 \bmod m$.

Доказательство. Используя неравенство Азумы (см., например, [50]), докажем, что случайно выбранное множество K будет "хорошим" с положительной вероятностью.

Пусть $1 \le l \le m-1$ и пусть K будет множеством t независимых случайных величин, равномерно распределенных по $\{0, \ldots, m-1\}$.

Определим случайные величины $X_i = \cos \frac{2\pi k_i l}{m}$ и $Y_k = \sum_{i=1}^{k} X_i$. Докажем, что неравенство Азумы применимо к последовательности $Y_0 = 0$, Y_1, Y_2, Y_3, ..., т.е. последняя является мартингалом с ограниченными разностями. Во-первых, необходимо доказать, что $E[Y_k] < \infty$.

Из определения X_i следует, что

$$E[X_i] = \frac{1}{m} \sum_{j=0}^{m-1} \cos \frac{2\pi jl}{m}.$$

Рассмотрим взвешенную сумму m-ых корней из единицы

$$\frac{1}{m} \sum_{j=0}^{m-1} \exp\left(\frac{2\pi jl}{m} i\right) = \frac{1}{m} \cdot \frac{\exp(2\pi ilm/m) - 1}{\exp(2\pi il/m) - 1} = 0,$$

т.к. l не кратно m.

$E[X_i]$ есть как раз действительная часть предыдущей суммы и поэтому равняется 0.

Следовательно, $E[Y_k] = \sum_{i=1}^{k} E[X_i] = 0 < \infty$.

Во-вторых, нужно показать, что условное ожидаемое значение следующего испытания при известных предыдущих исходах равняется последнему исходу.

$$E[Y_{k+1} | Y_1, \ldots, Y_k] = \frac{1}{m} \sum_{j=0}^{m-1} \left(Y_k + \cos \frac{2\pi jl}{m}\right) = Y_k + \frac{1}{m} \sum_{j=0}^{m-1} \cos \frac{2\pi jl}{m} = Y_k$$

Т.к. $|Y_{k+1} - Y_k| = |X_{k+1}| \le 1$ при $k \ge 0$, применение неравенства Азумы дает

$$Pr\big[|Y_t - Y_0| \ge \lambda\big] = Pr\left[\left|\sum_{i=1}^{t} X_i\right| \ge \lambda\right] \le 2 \exp\left(-\frac{\lambda^2}{2t}\right)$$

Поэтому вероятность того, что K не будет "хорошим" для $l \in [1, m-1]$ не превосходит

$$Pr\left[\left|\sum_{i=1}^{t} X_i\right| \ge \sqrt{\epsilon} t\right] \le 2 \exp\left(-\frac{\epsilon t}{2}\right) \le \frac{1}{m},$$

если $t = 2^{\lceil \log((2/\epsilon) \ln 2m) \rceil}$.

Таким образом, вероятность того, что построенное множество не будет "хорошим" хотя бы для одного $l \in [1, m-1]$ не больше $(m-1)/m < 1$. Следовательно, существует множество, которое будет "хорошим" для всех целых $l \in [1, m-1]$. Это множество также будет "хорошим" для всех $l \ne 0 \bmod m$, т.к. $\cos \frac{2\pi k(l+jm)}{m} = \cos \frac{2\pi kl}{m}$. □

Заметим, что можно было положить $t = \lceil (2/\epsilon) \ln 2m \rceil$, но для применения метода отпечатков нужно, чтобы t было степенью 2.

Конструктивные методы построения "хорошего" множества параметров Доказательство существования "хорошего" множества не дает конструктивного алгоритма его построения. Однако в работах [26], [44], [54] предлагаются конструктивные методы построения множеств с подобными свойствами.

Приведем вариант из [26]. Зафиксируем $\epsilon > 0$ и введем следующие обозначения:

$$P = \{p|\ p - \text{простое и } (\log m)^{1+\epsilon}/2 < p \leq (\log m)^{1+\epsilon}\},$$

$$S = \{1, 2, \ldots, (\log m)^{1+2\epsilon}\},$$

$$K = \{s \cdot p^{-1}|\ s \in S, p \in P\},$$

где p^{-1} – инвертирование по модулю m.

Очевидно, что $|K| = O(\log^{2+3\epsilon} m)$. При этом для любого целого $l \in \{1, 2, \ldots, m-1\}$ оказывается справедлива следующая оценка:

$$\frac{1}{|K|}\left|\sum_{k \in K} e^{\frac{2\pi k l}{m} i}\right| \leq (\log m)^{-\epsilon}.$$

Выделяя вещественную часть левой части неравенства и обозначая элементы множества K через $k_1, \ldots, k_t$, получаем:

$$\frac{1}{t}\left|\sum_{i=1}^{t} \cos\frac{2\pi k_i l}{m}\right| \leq (\log m)^{-\epsilon}$$

для любого $l \in \{1, 2, \ldots, m-1\}$.

Другими словами, множество оказывается "хорошим" для всех целых $l \neq 0 \bmod m$.

Описанный метод отпечатков обладает следующими свойствами:

- Он ориентирован на модели с классическим управлением, а значит и на квантовые ветвящиеся программы.
- Для рассматриваемых в данной работе функций образы входных наборов "легко" построить – используются только контролируемые вращения вокруг одной и той же оси на сходные углы и преобразования Адамара.
- Доказанная лемма гарантирует существование "хорошего" множества параметров, что позволяет ограничить вероятность ошибки сверху некоторой константой $\epsilon \in (0, 1)$.

Примеры Указанный подход демонстрируется на некоторых индивидуальных функциях, в основе которых лежит проверка равенства.

MOD_m Функция MOD_m проверяет, кратно ли число единиц во входном наборе параметру m. Линейный полином над $\mathbb{Z}_m$ выбран следующим образом:

$$\sum_{i=1}^{n} x_i.$$

MOD'_m Эта функция отличается от MOD_m только тем, что входной набор интерпретируется как двоичное число. Поэтому для данной функции можно использовать следующий полином над $\mathbb{Z}_m$:

$$\sum_{i=1}^{n} x_i 2^{i-1}.$$

EQ_n Функция EQ_n проверяет равенство двух n-битных двоичных наборов и может задаваться следующим полиномом над $\mathbb{Z}_{2^n}$:

$$\sum_{i=1}^{n} x_i 2^{i-1} - \sum_{i=1}^{n} y_i 2^{i-1}.$$

$Palindrome_n(x_1, \ldots, x_n) \equiv \left[x_1 x_2 \ldots x_{\lfloor n/2 \rfloor} = x_n x_{n-1} \ldots x_{\lceil n/2 \rceil + 1}\right]$. Для данной функции существует следующий полином над $\mathbb{Z}_{2^{\lfloor n/2 \rfloor}}$:

$$\sum_{i=1}^{\lfloor n/2 \rfloor} x_i 2^{i-1} - \sum_{i=\lceil n/2 \rceil}^{n} x_i 2^{n-i}$$

$PERM_n$ Эта функция проверяет, является ли булевская $n \times n$ матрица перестановочной, т.е. содержащей ровно одну единицу в каждой строчке и каждом столбце. Для данной функции существует следующий линейный полином над $\mathbb{Z}_{(n+1)^{2n}}$:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} \left((n+1)^{i-1} + (n+1)^{n+j-1}\right) - \sum_{i=1}^{2n} (n+1)^{i-1}.$$

Для описанных функций в таблице приводятся нижние оценки сложности (ширины) реализации в детерминированных OBDD и получаемые по нашему методу верхние оценки сложности представления в квантовых OBDD.

	OBDD	QOBDD
MOD_m	$\Omega(m)$	$O(\log m)$
MOD'_m	$\Omega(m)$	$O(\log m)$
EQ_n	$2^{\Omega(n)}$	$O(n)$
$Palindrome_n$	$2^{\Omega(n)}$	$O(n)$
$PERM_n$	$\Omega(2^n n^{-1/2})$	$O(n \log n)$

2.2.2 Метод Баррингтона

Известный результат статьи [4] позволяет моделировать схемы из класса $\mathbf{NC^1}$ ветвящимися программами константной ширины, используя коммутаторы подстановок из группы S_5. В работе [6] была проанализирована структура получаемых программ и показано, что они являются k-OBDD, где $k = n^{O(1)}$. В статье [17] предлагается моделирование конструкции Баррингтона на 1 кубите в квантовых ветвящихся программах, что позволяет представить любую функцию из $\mathbf{NC^1}$ 1-кубитной квантовой $k(n)$-OBDD.

Баррингтоном была предложена следующая конструкция: произвольной схеме из конъюнкторов и инверторов ставилась в соответствие перестановочная ветвящаяся программа, состоящая из последовательности инструкций, выдающих одну из двух подстановок из $A_5 \in S_5$ в зависимости от значения считываемой переменной. Такая программа выдает ответ 0, если композиция выданных подстановок есть тождественная подстановка $id \in A_5$, и единицу в противном случае. В основе индуктивного построения такой программы лежит моделирование конъюнкции при помощи четырех инструкций следующего вида: $\langle x_{i_1}, \tau_1, id\rangle$, $\langle x_{i_2}, \tau_2, id\rangle$, $\langle x_{i_1}, \tau_1^{-1}, id\rangle$, $\langle x_{i_2}, \tau_2^{-1}, id\rangle$. Другими словами, при $x_{i_1} = x_{i_2} = 1$ выдается последовательность подстановок $\tau_1\tau_2\tau_1^{-1}\tau_2^{-1} \neq id$, а в остальных случаях выдается последовательность, вырождающаяся в тождественную подстановку. Для моделирования отрицания необходимо лишь модифицировать последнюю инструкцию программы.

Для метода Баррингтона существенным моментом является использование неразрешимой группы, что позволяет на каждом шаге выбирать такие элементы τ_1 и τ_2, что их коммутатор $\tau_1\tau_2\tau_1^{-1}\tau_2^{-1}$ не вырождается в тождественную подстановку.

Данный результат обобщается на случай произвольной неабелевой группы. В частности, программы над группой двумерных унитарных преобразований также могут вычислять все функции из $\mathbf{NC^1}$, что и было доказано в статье [17].

Глава 3

Математическое описание класса задач, эффективно решаемых квантовыми ветвящимися программами

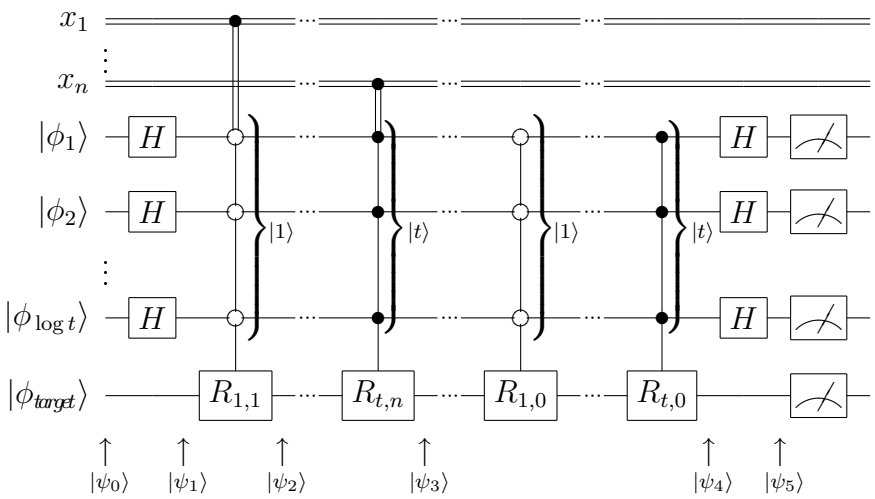
3.1 Эффективные алгоритмы, основанные на методе отпечатков

На основе предложенного нами метода отпечатков, описанного в предыдущем разделе, можно сформулировать достаточное условие эффективной вычислимости булевых функций в модели один раз читающих квантовых ветвящихся программ. Тем самым описывается некоторый класс задач, эффективно решаемых в указанной модели.

Пусть g – есть характеристический полином булевой функции $f(x_1, \ldots, x_n)$. Справедлива следующая теорема.

Теорема 3.1.1. Пусть $\epsilon \in (0, 1)$. Если g является линейным полиномом на кольцом $\mathbb{Z}_m$, т.е. $g = c_1x_1 + \ldots c_nx_n + c_0$, то f может быть вычислена с односторонней ошибкой ϵ квантовой OBDD ширины $O\left(\frac{\log m}{\epsilon}\right)$.

Доказательство. Приведем алгоритм в виде схемы:



Исходное состояние квантового регистра $|\psi_0\rangle = |\phi_1\rangle \otimes |\phi_2\rangle \otimes \cdots \otimes |\phi_{\log t}\rangle \otimes |\phi_{target}\rangle = |0\rangle^{\otimes \log t} |0\rangle$.

Контролируемые унитарные операторы $R_{i,j}$ для $i \in \{1,\ldots,t\}$, $j \in \{0,\ldots,n\}$ задаются следующим образом:

$$R_{i,j} = R_{\hat{y}}\left(\frac{4\pi k_i c_j}{m}\right).$$

Здесь c_j – это коэффициенты линейного полинома для f, а множество параметров $K = \{k_1,\ldots,k_t\}$ является "хорошим" согласно технике отпечатков из [23] при $t = 2^{\lceil \log((2/\epsilon)\ln 2\cdot m)\rceil}$.

Пусть $\sigma = \sigma_1 \ldots \sigma_n \in \{0,1\}^n$ – это входной двоичный набор.

Первый слой преобразований H переводит $|\psi_0\rangle$ в следующую суперпозицию:

$$|\psi_1\rangle = \frac{1}{\sqrt{t}} \sum_{i=1}^{t} |i\rangle\,|0\rangle\,.$$

Далее, если очередной входной символ есть 0, то применяется тождественное преобразование I. В противном случае, считав очередное значение $\sigma_j = 1$, состояние последнего кубита преобразуется оператором $R_{i,j}$, поворачивающим его на угол, пропорциональный c_j. Более того, такой поворот осуществляется в каждом из t подпространств с соответствующей амплитудой $1/\sqrt{t}$. Такой квантовый параллелизм реализуется контролируемыми операторами $C_i(R_{i,j})$, преобразующими состояния вида $|i\rangle\,|\cdot\rangle$ в $|i\rangle\,R_{i,j}\,|\cdot\rangle$ и оставляющими прочие состояния без изменений. Например, считав входной символ $\sigma_1 = 1$, система перейдет в состояние

$$\begin{aligned} |\psi_2\rangle &= \tfrac{1}{\sqrt{t}} \sum_{i=1}^{t} C_i(R_{i,1})\,|i\rangle\,|0\rangle = \tfrac{1}{\sqrt{t}} \sum_{i=1}^{t} |i\rangle\,R_{i,1}\,|0\rangle \\ &= \tfrac{1}{\sqrt{t}} \sum_{i=1}^{t} |i\rangle \left(\cos\tfrac{2\pi k_i c_1}{m}\,|0\rangle + \sin\tfrac{2\pi k_i c_1}{m}\,|1\rangle\right) \end{aligned}.$$

Таким образом, после считывания входного набора σ в амплитудах базисных состояний "накопится" сумма $\sum_{j=1}^{n} c_j\sigma_j$:

$$|\psi_3\rangle = \tfrac{1}{\sqrt{t}} \sum_{i=1}^{t} |i\rangle \left(\cos\tfrac{2\pi k_i \sum_{j=1}^{n} c_j\sigma_j}{m}\,|0\rangle + \sin\tfrac{2\pi k_i \sum_{j=1}^{n} c_j\sigma_j}{m}\,|1\rangle\right).$$

На следующем шаге применяются вращения на углы, пропорциональные c_0. Тем самым состояние регистра переводится в

$$|\psi_4\rangle = \tfrac{1}{\sqrt{t}} \sum_{i=1}^{t} |i\rangle \left(\cos\tfrac{2\pi k_i g(\sigma)}{m}\,|0\rangle + \sin\tfrac{2\pi k_i g(\sigma)}{m}\,|1\rangle\right).$$

Следовательно, применив преобразование $H^{\otimes \log t} \otimes I$, получаем состояние

$$\begin{aligned} |\psi_5\rangle &= \left(\tfrac{1}{t}\sum_{i=1}^{t} \cos\tfrac{2\pi k_i g(\sigma)}{m}\right) |0\rangle^{\otimes \log t}\,|0\rangle + \\ &\quad +\gamma\,|0\rangle^{\otimes \log t}\,|1\rangle + \sum_{i=2}^{t} |i\rangle\,(\alpha_i\,|0\rangle + \beta_i\,|1\rangle)\,, \end{aligned}$$

где γ, α_i и β_i – некоторые амплитуды, значения которых для нас не важны.

Входной набор принимается, если результатом измерения регистра оказывается состояние $|0\rangle^{\otimes \log t}\,|0\rangle$. Значит, вероятность принятия есть

$$Pr_{accept}(\sigma) = \frac{1}{t^2}\left(\sum_{i=1}^{t} \cos\frac{2\pi k_i g(\sigma)}{m}\right)^2.$$

Если $f(\sigma) = 1$, то $g(\sigma) = 0$, и набор σ принимается с вероятностью 1. В противном случае, выбор множества $K = \{k_1, \ldots, k_t\}$ гарантирует, что

$$Pr_{accept}(\sigma) = \frac{1}{t^2}\left(\sum_{i=1}^{t} \cos\frac{2\pi k_i g(\sigma)}{m}\right)^2 < \epsilon.$$

Таким образом, f может быть вычислена q-кубитной квантовой OBDD, где $q = \log 2t = O(\log\log m)$. Соответственно, ширина программы есть $2^q = O(\log m)$. □

Примерами функций, обладающих линейными характеристическими полиномами, являются MOD_m, MOD'_m, EQ_n, $Palindrome_n$, $Period^s_n$, $Semi - Simon^s_n$, $PERM_n$ и т.п.

3.2 Эффективное вычисление функций из класса $\mathbf{NC^1}$

Известный класс $\mathbf{NC^1}$ содержит функции, представимые классическими схемами из функциональных элементов логарифмической глубины и полиномиальной сложности. Данный класс является достаточно широким, неизвестно даже собственным ли образом он входит в класс **NP**.

В работе [17] доказано, что все функции из данного класса могут быть вычислены квантовыми ветвящимися программами ширины 2, т.е. с использованием всего лишь одного кубита. Позже, в статье [6] была уточнена структура этих ветвящихся программ, и доказано, что они являются k-раз читающими квантовыми ветвящимися программами с порядком считывания переменных, основанном на конструкции Баррингтона [4].

Таким образом, квантовые ветвящиеся программы константной ширины и полиномиальной длины могут эффективно решать любые задачи из класса $\mathbf{NC^1}$.

Литература

[1] Аблаев Ф.М. Влияние степени изолированности точки сечения на число состояний вероятностного автомата / Ф.М. Аблаев // Математические заметки. – М.:Изд-во "Наука Главная редакция физико-математической литературы, 1988. – Т. 4, N 3. – С. 289-297.

[2] Аблаев, Ф.М. О вычислениях в квантовых ветвящихся программах методом “характерных признаков” / Ф.М. Аблаев, А.В. Васильев // Материалы XV Международной конференции Проблемы теоретической кибернетики (Казань, Россия, 2Ц7 июня, 2008). – Казань: Изд-во “Отечество”, 2008. – С. 1.

[3] Александров П.С. Введение а теорию множеств и общую топологию / П.С. Александров. - М.: Главная редакция физико-математической литературы издательства "Наука 1977. – 368 с.

[4] Баррингтон, Д. Ветвящиеся программы ограниченной ширины, имеющие полиномиальную сложность, распознают в точности языки из NC^1 / Д. Баррингтон // Кибернетический сборник. – М.: Мир, 1991. – Вып. 28. – С. 94-113.

[5] Бухараев Р.Г. Основы теории вероятностных автоматов / Р.Г. Бухараев. – М.: Наука. Главная редакция физико-математической литературы, 1985. – 288 с.

[6] Васильев, А.В. О функциях, вычислимых булевыми схемами логарифмической глубины и ветвящимися программами специального вида / А.В. Васильев // Дискретный анализ и исследование операций. – Серия 1. – 2007. – Т.14, вып. 3. – С. 31Ц39.

[7] Гайнутдинова, А.Ф. О сравнительной сложности квантовых и классических бинарных программ / А.Ф. Гайнутдинова // Дискретная математика. – М.: Изд-во РАН, 2002. – Т.14, вып. 3. – С. 109-121.

[8] Нигматуллин, Р.Г. Сложность булевых функций, Издательство Наука, Москва.

[9] Нильсен, М. Квантовые вычисления и квантовая информация / М. Нильсен, И. Чанг; Пер. с англ. под ред. М.Н. Вялого и П.М. Островского с предисловием К.А. Валиева. – М.: Мир, 2006. – 824 с.

[10] Покровская И.А. Некоторые оценки числа состояний вероятностных автоматов, представляющих регулярные события / И.А. Покровская // Проблемы кибернетики. – М.: Изд-во физ.-мат литературы, 1979. – вып. 36. – С. 181-194.

[11] Рабин М. Вероятностные автоматы / М. Рабин // Кибернетический сборник. – М.: Изд-во Мир, 1964. – вып.9. – С. 123-141.

[12] Фрейвалд Р.В. Об увеличении числа состояний при детерминизации конечных вероятностных автоматов / Р.В. Фрейвалд // Автоматика и вычислительная техника, 1982. – N 3. – С. 39-42.

[13] Яблонский, С.В. Введение в дискретную математику / С.В. Яблонский. – М: Наука, 1986. – 384 с.

[14] Aaronson, S. The Complexity Zoo / S. Aaronson // http://www.complexityzoo.com. – 2004.

[15] Ablayev, F. On computational power of quantum branching programs / F. Ablayev, A. Gainutdinova, M. Karpinski // Lecture Notes in Computer Science. – Springer-Verlag, 2001. – V. 2138. – P. 59-70.

[16] Farid Ablayev, Aida Gainutdinova. Classical Simulation Complexity of Quantum Machines. *Fundamentals of Computation theory: 14th International Symposium; proceedings/ FCT 2003, Malmo, Sweden, August 2003, Lecture Notes in Computer Science.* (2003) 2751, 296-302.

[17] On the computational power of probabilistic and quantum branching programs of constant width / F. Ablayev, A. Gainutdinova, M. Karpinski, C. Moore, C. Pollette // Information and Computation. – Elsevier, 2005.

[18] Ablayev, F. On the power of randomized ordered branching programs / F. Ablayev, M. Karpinski // Electronic Colloquium on Computational Complexity (http://www.eccc.uni-trier.de/eccc/). – TR98-004, 1998.

[19] Ablayev, F. On the power of randomized branching programs / F. Ablayev, M. Karpinski // Proc. 28th ICALP (1996). – LNCS, Springer, 1996. – V. 1099. – P. 348-356.

[20] Ablayev, F. A Lower Bound for Integer Multiplication on Randomized Read-Once Branching Programs / F. Ablayev, M. Karpinski // Information and Computation. – Elsevier, 2003. – V. 186, N 1. – P. 78-89.

[21] Ablayev, F.M. On Complexity of Quantum Branching Programs Computing Equality-like Boolean Functions / F.M. Ablayev, A.F. Khasianov, A.V. Vasiliev // Electronic Colloquium on Computational Complexity (http://www.eccc.uni-trier.de/eccc/). – TR08-085, 2008.

[22] Ablayev, F. Quantum and Stochastic Branching Programs of Bounded Width / F. Ablayev, C. Moore, C. Pollette // Proc. of the ICALP'2002, Lecture Notes in Computer Science. – Springer-Verlag, 2002. – P. 343-354.

[23] Ablayev, F.M. On the Computation of Boolean Functions by Quantum Branching Programs via Fingerprinting / F.M. Ablayev, A.V. Vasiliev // Electronic Colloquium on Computational Complexity (http://www.eccc.uni-trier.de/eccc/), TR08-059, 2008.

[24] L. Adleman, J. Demarrais, M. Huang, Quantum computability, SIAM J. on Computing. 26(5), (1997), 1524–1540.

[25] Adleman, L. Quantum computability / L. Adleman, J. Demarrais, M. Huang // SIAM J. on Computing. – 1997. – V. 26, N 5. – P. 1524-1540.

[26] Construction of a thin set with small Fourier coefficients / M. Ajtai, H. Iwaniec, J. Komlos, J. Pintz, E. Szemeredi // Bulletin of the London Mathematical Society. – 1990. – V. 22. – P. 583-590.

[27] Freivalds, R. 1-way quantum finite automata: strengths, weaknesses and generalization / R. Freivalds, A. Ambainis // Proceeding of the 39th IEEE Conference on Foundation of Computer Science. – 1998. – P. 332-342.

[28] Ambainis, A., Nahimovs N. Improved constructions of quantum automata / A. Ambainis, N. Nahimovs // http://xxx.lanl.gov/archive/quant-ph. – arXiv:0805.1686v1, 2008.

[29] J.L. Balcazar, J. Diaz, J. Gabarro, Structural Complexity 1. - Springer-Verlag Berlin Heidelberg 1988, 1995.

[30] Barrington, D.M. Finite monoids and the fine structure of NC^1 / D.M. Barrington, D. Thérien // Journal of the ACM. – 1988. – V. 35. – P. 941-952.

[31] Barrington, D. Lectures on Computational Complexity Theory / D.M. Barrington, A. Maciel // IAS/Park City Mathematics Institute 2000 Summer Session. – http://people.clarkson.edu/ alexis/PCMI. – 2000.

[32] Bennett, C.H. Logical reversibility of computations / C.H. Bennett // IBM Jounal of Res. Develop. – 1973. – V. 17. – P. 525-532.

[33] Benioff, P.A. Quantum mechanical hamiltonian models of turing machines / P.A. Benioff // Journal of Statistical Physics. – 1982. – V. 29, N 3. – P. 515-546.

[34] Bernstein, E. Quantum complexity theory / E. Bernstein, U. Vazirani // SIAM J. Comput. – 1997. – V. 26, N 5. – P. 1411-1473.

[35] Bryant, R. Symbolic boolean manipulation with ordered binary decision diagrams / R. Bryant //ACM Computing Surveys. – V. 24, No. 3. – 1992. – P. 293-318.

[36] Quantum fingerprinting / H. Buhrman, R. Cleve, J. Watrous J., R. de Wolf // Physical Review Letters. – 2001. – 87(16):167902.

[37] Bryant, R. On the complexity of VLSI implementations and graph representations of Boolean functions with applications to integer multiplication / R. Bryant // IEEE Trans. Comput. – 40 (2). – 1991. – P. 205-213.

[38] Deutsch, D. Quantum theory, the Church-Turing principle and the universal quantum computer / D. Deutsch // Proccedings of the Royal Society. - London, 1985. – A400. – P. 97-117.

[39] Deutsch, D. Rapid solution of problems by quantum computation / D. Deutsch, R. Jozsa // Proc. of the Royal Society. – London, 1992. - A439. – P. 553-558.

[40] Feynman, R. Simulating physics with computers / R. Feynman // International Journal of Theoretical Physics. – 1982. – V. 21, N 6,7. - P. 467-488.

[41] Furst, M. Parity, circuits, and the polynomial-time hierarchy / M. Furst, J.B. Saxe, M. Sipser // Math. Systems Theory. – 1984. – 17:13Ц27.

[42] Grover, L. A fast quantum mechanical algorithm for database search / L. Grover // Proc. of 28th STOC, 1996. – P.Philadelphia PA USA, 2996. – P. 212-219.

[43] Jukna S. On the effect of null-chains on the complexity of constant schemes // Proc. FCT'89, Lecture Notes in Computer Science. – V. 380. – Springer-Verlag, 1989. – P. 246-256.

[44] Katz, N.M. An Estimate for Character Sums / N.M. Katz // Journal of the American Mathematical Society. – 1989. – P. 197-200.

[45] Khasianov, A. Complexity Bounds On Some Fundamental Computational Problems For Quantum Branching Programs / A. Khasianov. – Ph.D. thesis, University of Bonn. – http://nbn-resolving.de/urn:nbn:de:hbz:5N-05696.

[46] Kondacs A. On the power of quantum finite state automata / A. Kondacs, J. Watrous // Proc. of the 38th Annual Symposium on Foundations of Computer Science, 1997. – P. 66-75.

[47] Krause, M. Separating the Eraser Turing Machine Classes L_e, NL_e, $co - NL_e$, and P_e / M. Krause, C. Meinel, S. Waack // Proceedings MFCS'88, Lecture notes in computer science. – Springer-Verlag, 1988. – V. 324. – P. 405-413.

[48] Lee, C.Y. Representation of switching circuits by binary-decision programs / C.Y. Lee // Bell Sysytems Technical Journal. – 39. – 1959. – P. 985-999.

[49] Moore, C. Quantum automata and quantum grammars / C. Moore, J. Crutchfield // Theoretical Computer Science. – 2000. – 237. – P. 275-306.

[50] Motwani R. Randomized Algorithms / R. Motwani, P. Raghavan. – Cambridge University Press, 1995. – 492 p.

[51] Nakanishi, M. Ordered quantum branching programs are more powerful than ordered probabilistic branching programs under a bounded-width restriction / M. Nakanishi, K. Hamaguchi, T. Kashiwabara // Proc. of the 6th Annual International Conference on Computing and Combinatorics, COCOON'2000. – Lecture Notes in Computer Science, Springer-Verlag, 2000. – V. 1858. – P. 467-476.

[52] Ponzio, S. Restricted branching programs and hardware verification / S. Ponzio. – Ph.D. thesis, MIT, 1995. - http://www.eccc.uni-trier.de/eccc/

[53] Pudlák, P. A Lower Bound on Complexity of Branching Programs (Extended Abstract) // Proceedings of the Mathematical Foundations of Computer Science. – Springer-Verlag, 1984. – P. 480–489.

[54] Razborov, A. Constructing small sets that are uniformin arithmetic progressions / A. Razborov, E. Szemeredi, A. Wigderson // Combinatorics, Probability and Computing. – 1993. – V.2. – P. 513Ц518.

[55] Sauerhoff M. A lower bound for randomized read-k-times branching programs // ECCC, TR97-019. – 1997. Доступно на http://www.eccc.uni-trier.de/eccc/.

[56] Sauerhoff, M. Quantum branching programs and space-bounded nonuniform quantum complexity / M. Sauerhoff, D. Sieling // http://xxx.lanl.gov/archive/quant-ph. – ph/0403164. – 2004.

[57] Sauerhoff, M. Quantum vs. Classical Read-Once Branching Programs // arXiv:quant-ph/0504198 v1. – 2005.

[58] Shor, P. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer / P. Shor // SIAM J. on Computing. – 1997. – V. 26, N 5. – P. 1484-1509.

[59] Simon, D. On the power of quantum computation / D. Simon // SIAM Journal of Computing. – 1997. – V. 26, N 5. – P. 1474-1483.

[60] Turakainen P. Generalized automata and stochastic languages / P. Turakainen // Proc. Amer. Math. Soc.,1969. – V. 21. – P. 303-309.

[61] Vasiliev, A.V. Functions computable by Boolean circuits of logarithmic depth and branching programs of a special type / A.V. Vasiliev // Journal of Applied and Industrial Mathematics. – 2008. – Vol. 2. – No. 4. – P. 585-590.

[62] J. Watrous. Space-bounded quantum complexity. Journal of Computer and System Sciences, 59(2), (1999), 281–326.

[63] Wegener, I. The Complexity of Boolean Functions / I. Wegener. – Stuttgart: John Wiley & Sons Ltd, and B. G. Teubner, 1987. – 458 p.

[64] Wegener, I. Efficient data structures for boolean functions / I. Wegener. – Discrete Mathematics. – 1994. – Vol. 136. – P. 347-372.

[65] Wegener, I. Branching programs and binary decision diagrams / I. Wegener. – SIAM Monographs on Discrete Mathematics and Applications, SIAM Press, 2000. – 409 p.

[66] Yao, A. Some complexity questions related to distributive computing / A. Yao // Proceedings of 11th STOC. – 1979. – P. 209-213.

[67] Yao A. Quantum circuit complexity / A. Yao // Proc. 34th IEEE Symposium on Foundation of Computer Science. – 1993. – P. 352-361.

[68] Китаев А., Шень А., Вялый М. Классические и квантовые вычисления// М.: МЦНМО, 1999. – 192 с.

[69] Соловьев Н.А. Тесты (теория, построение, применение). — Новосибирск, Наука, 1978.

[70] Фрейвалд Р.В. Ускорение распознавания некоторых множеств применением датчика случайных чисел. — Проблемы Кибернетики, 1979, вып. 36, с. 209-224.

[71] Холево А. С. Квантовые системы, каналы, информация // Москва. : МЦНМО, 2010. – 327 с.

[72] F. Ablayev, Lower Bounds for Probabilistic Space Complexity: Automata Approach, *TR423, May 1992, 13p, University of Rochester USA.*

[73] F. Ablayev, Lower Bounds for one-way Probabilistic Communication Complexity, *in Proceedings of the ICALP'93, Lecture Notes in Computer Science, Springer-Verlag,* 1993, 700, 241-252. See also: *TR422, May, 1992, University of Rochester USA.*

[74] F. Ablayev, Lower bounds for probabilistic space complexity: communication automata approach, *in Proceedings of the LFCS'94, Lecture Notes in Computer Science, Springer-Verlag*, 1994, 813, 1-7.

[75] F. Ablayev, Lower bounds for one-way probabilistic communication complexity and their application to space complexity, *Theoretical Computer Science,* 1996, v.157, 139-149.

[76] Farid Ablayev and Aida Gainutdinova. On the Lower Bound for One-Way Quantum Automata. *Proceedings of MFCS'2000, LNCS 1893, Springer*, pp.132-140, 2000.

[77] Ablayev F., Gainutdinova A. Complexity of quantum uniform and nonuniform automata. // Proc of the Interantial Confeence Developments in Language Theory (DLT 2005), Palermo; proceeding, Lecture Notes in Computer Science, Springer-Verlag, 2005, 3572, p.78-87.

[78] D. Aharonov, A. Kitaev, and N. Nisan, Quantum circuits with mixed states. In *Proc. of 30th STOC*, 20-30, 1998.

[79] A.Alu, N.Engheta arXiv:0805.1757v1 physics/optics.

[80] A. Ambainis, L. Schulman, and U. Vazirani, Computing with highly mixed states. *Proc. 32nd ACM Symp. on Theory of Computing* 697–704, 2000.

[81] Barenco, A., Bennett, C.H., Cleve, R, DiVincenzo, D.P., Margolus, N., Shor, P., Sleater, T., Smolin, J., Weinfurter, H.: *Phys. Rev. A* **52**, (1995) 3457.

[82] Boneh, D., and Lipton, R.J.: Quantum cryptanalysis of hidden linear functions (Extended abstract). Lecture Notes on Computer Science **963** (1995) 424–437.

[83] G. Brassard. Quantum communication complexity (a survey). quant-ph/0101005, 1 January 2001.

[84] A. Brodsky and N. Pippenger, Characterizations of 1-way quantum finite automata, quant-ph/9903014, 1999.

[85] Cleve, R., Ekert, E., Macchiavello, C., and Mosca, M.: Quantum Algorithms Revisited, Proc. Roy. Soc. Lond. A, **454**, (1998) 339-354.

[86] T. Cover and J. Thomas. elements of information theory, John Wiley & Sons, Inc., 1991.

[87] Deutsch, D. : Quantum Theory, the Church-Turing principle and the universal quantum computer. Proc. Roy. Soc. Lond. A, **400**, (1985) 97-117.

[88] R.A.Depine, M.L.M.Ricci, J.A.Monsoriu, E.Silvestre, P. Andres arXiv:0606069v1 physics/optics.

[89] N.Garcia, M.Munoz, E.V.Ponizovskaya, M.Nieto-Vesperinas arXiv: 0206476v1 cond/mat.

[90] Grigoriev, D. Y.: Testing the shift-equivalence of polynomials by deterministic, probabilistic and quantum machines. Theoretical Computer Science, **180** (1997) 217-228.

[91] M. P. Hedges, M. J. Sellars, Y.-M. Lee, J. J. Longdell. In a poster presented at Int. Conf. Hole Burning, Single Molecule and Related Spectroscopies: Science Applications, 22Ц27 June 2009.

[92] G. Hetet et al., Opt. Lett. **33**, 2323 (2008)

[93] J. Hromkovich. Communication complexity and parallel computing, *Springer-Verlag, Berlin, Heidelberg, New York*, 1997.

[94] Peter Høyer, *Conjugated operators in quantum algorithms*, Tech. report, University of Southern Denmar, 1997.

[95] Ilan Kremer, Noam Nisan, Dana Ron: On randomized one-round communication complexity. In *Proceedings of the 27th annual ACM symposium on Theory of computing*, 1995, 596-605.

[96] Jozsa, R.: Quantum Algorithms and the Fourier Transform, Proc. Roy. Soc. Lond. A, **454**, (1998) 323-337.

[97] H. Klauck. On quantum and probabilistic communication: La Vegas and one-way protocols. *In the Proc. of the 32nd ACM Symp. Theory of Computing*, 2000.

[98] H. Klauck. Quantum communication complexity. quant-ph/0005032, 2000. Available at `http://xxx.itep.ru`.

[99] P. Kok et. al., Rev. Mod. Phys. **79**, 135 (2007).

[100] A. Kondacs, J. Watrous, On the power of quantum finite state automata. In *Proceeding of the 38th IEEE Conference on Foundation of Computer Science*, 1997, 66-75.

[101] E. Kushilevitz and N. Nisan. Communication Complexity, *Cambridge University Press*, 1997.

[102] Kitaev, A. Y. : Quantum measurements and the Abelian stabiliser problem. e-print quant-ph/9511026 (1995)

[103] I. Kremer. Quantum communication. Master's thesis, Hebrew University, Jerusalem, 1995.

[104] Lovasz L. Communication Complexity: A Survey. — in "Paths, Flows and VLSI Layout", Korte, Lovasz, Promel, Schrijver Eds., Springer-Verlag 1990, p. 235-266.

[105] A.I.Lvovsky, B.C. Sanders, W. Tittel. Optical quantum memory // Nature Photonics, **30**, 706 (2009).

[106] M. Lukin and A. Imamoglu, Nature (Lond.) **413**, 273 (2001).

[107] M. Mosca and A. Ekert, *The hidden subgroup problem and eigenvalue estimation on a quantum computer*, arXive e-print quant-ph/9903071, 1999.

[108] A. Nayak, Optimal lower bounds for quantum automata and random access codes, *Proceeding of the 40th IEEE Conference on Foundation of Computer Science*, 1999, 369-376. See also quant-ph/9904093

[109] J.Nunn et.al., Phys.Rev.Lett. **101**, 260502 (2008).

[110] M.O. Rabin, Probabilistic automata, *Information and Control*, 6 (1963), 230-244.

[111] A. Razborov. Quantum communication complexity of symmetric predicates. *Izvestia of the Russian Academy of Science, mathematics*, vol. 67, No 1, 2003, 159-176. English version avaliable at `http://www.mi.ras.ru/~razborov/`

[112] H. Schmidt and A. Imamoglu, Opt. Lett. **21**, 1936 (1996).

[113] Z.-B. Wang, K.-P. Marzlin and B. C. Sanders Phys. Rev.Lett. **97**, 063901 (2006).

[114] R. de Wolf. Quantum communication and complexity. *Theoretical Computer Science*, 287, (1), 337-353, 2002

[115] A. Yao. Some Complexity Questions Related to Distributive Computing. In *Proceedings of the 11th Annual ACM Symposium on the Theory of Computing*, (1979), 209-213.

[116] Yao A. Lower Bounds by Probabilistic Arguments. — in Proc. of the 24th IEEE Symposium on Foundations of Computer Science, 1983, p. 420-428.