УДК 004

Д.А. Селицкий, мл. науч. сотр.
(Институт физики НАН Беларуси, г. Минск)

СРАВНЕНИЕ ЭФФЕКТИВНОСТИ СИМУЛЯЦИИ АЛГОРИТМА ГРОВЕРА И КЛАССИЧЕСКИХ АЛГОРИТМОВ ПОИСКА

В недавних исследованиях [1] было показано, что классические симуляторы квантовых вычислений при моделировании алгоритма Гровера [2] требуют до 16 ГБ оперативной памяти уже при 30 кубитах, что ставит под сомнение практическую применимость теоретического квадратичного ускорения поиска на классическом оборудовании. Этот факт подчёркивает разрыв между асимптотическими оценками квантовых алгоритмов и их реальной вычислительной стоимостью при классической симуляции.

Актуальность данной работы обусловлена активным использованием классических симуляторов в исследованиях квантовых алгоритмов на фоне отсутствия масштабируемых квантовых процессоров. Несмотря на обширную теоретическую базу алгоритма Гровера, современные исследования [3] указывают на экспоненциальные ограничения симуляции, которые могут нивелировать ожидаемое преимущество по сравнению с классическими алгоритмами поиска.

Цель работы – провести экспериментальное сопоставление вычислительных затрат при реализации алгоритма Гровера на классическом симуляторе на языке Q# и классического алгоритма поиска на языке C++.

Для корректного сравнения квантового и классического алгоритма поиска будем использовать одинаковое число элементов в неупорядоченной базе данных N = 2n, где n варьируется от 10 до 25. Каждая конфигурация будет включать один элемент, который необходимо будет найти с использованием алгоритма Гровера и классического алгоритма поиска.

Схема алгоритма Гровера представлена на рис. 1. При реализации алгоритма Гровера будем задавать следующее число итераций [4]:

$$N_{opt} = \frac{\pi}{4}\sqrt{\frac{N}{M}}, \qquad (1)$$

где N_{opt} – оптимальное число итераций, $N = 2^n$ – число элементов в неупорядоченной базе данных, M – количество решений задачи поиска.

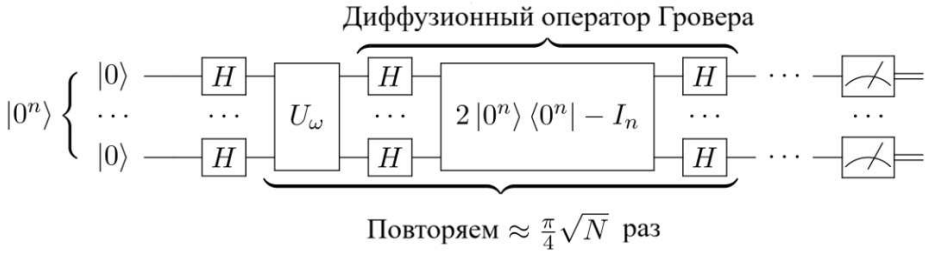

Рисунок 1 – Схема алгоритма Гровера

В качестве классического алгоритма поиска был выбран алгоритм параллельного поиска [5]. Блок-схема данного алгоритма представлена на рис. 2.

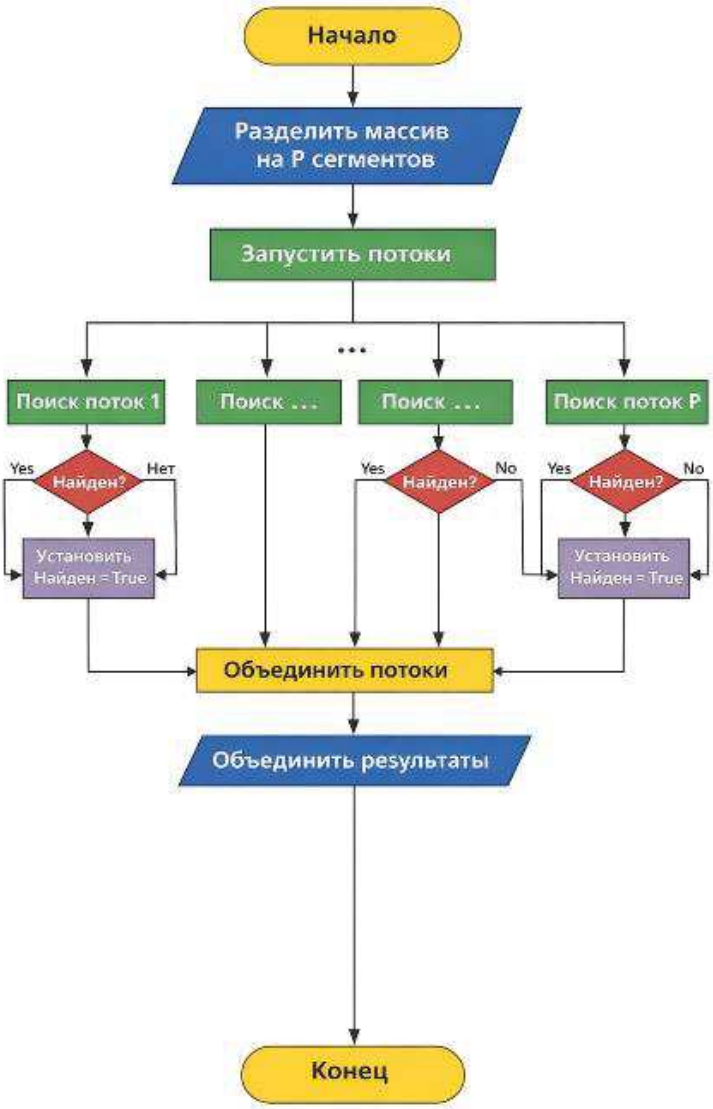

Рисунок 2 – Блок-схема алгоритма параллельного поиска

Особенностью данного алгоритма является одновременный поиск в нескольких потоках. Исходный массив данных разбивается на P непересекающихся сегментов, где P – количество используемых потоков. Для каждого сегмента запускается отдельный поток, который независимо выполняет поиск заданного элемента в своём диапазоне данных. Поиск внутри каждого потока осуществляется последовательно и может быть остановлен досрочно, если искомый элемент найден. После завершения всех потоков их результаты объединяются, и алгоритм возвращает информацию о наличии элемента и, при необходимости, его позицию. Время выполнения данного алгоритма может быть оценено как O(N/P).

Результаты реализации данных алгоритмов представлены ниже в таблице и на рис. 3. Основными критериями сравнения являлись время нахождения искомого элемента, а также объём памяти, задействованный алгоритмом при решении задачи поиска.

Таблица – Сравнение эффективности алгоритма Гровера, реализованного на языке Q#, и классического алгоритма параллельного поиска, реализованного на языке программирования C++

Число элементов в неупорядоченной базе данных	Параллельный поиск на C++	Алгоритм Гровера на Q#
$2^{10} = 1024$	0,0004 с / 3,89 МБ	0.01 с / 0,02 МБ
$2^{13} = 8192$	0,0005 с / 3,92 МБ	2.7 с / 0,12 МБ
$2^{15} = 32768$	0,0006 с / 3,86 МБ	15 с / 0,5 МБ
$2^{17} = 131072$	0,0014 с / 3,89 МБ	47 с / 2 МБ
$2^{20} = 1048576$	0,0090 с / 7,56 МБ	161 с / 16 МБ
$2^{22} = 4194304$	0,0342 с / 19,7 МБ	322 с / 64 МБ
$2^{25} = 33554432$	0,2750 с / 134 МБ	910 с / 512 МБ

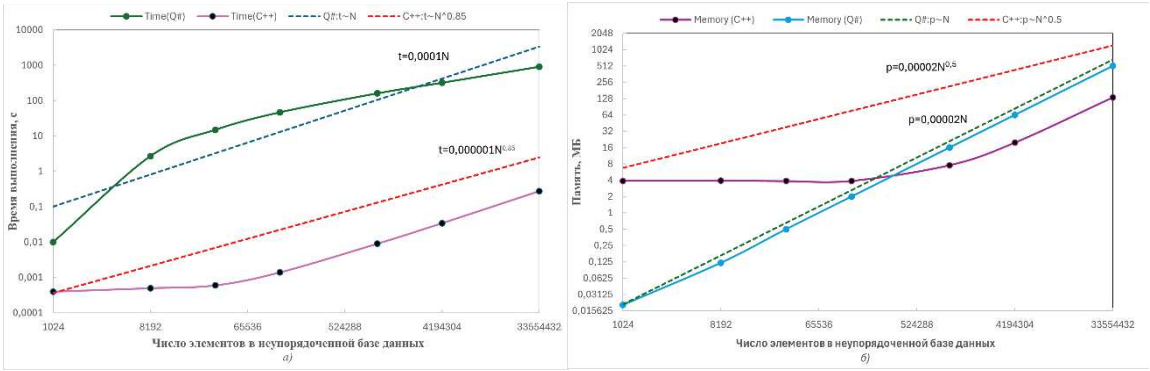

Рисунок 3 – Сравнение алгоритма параллельного поиска на C++ и симуляции алгоритма Гровера на Q# в зависимости от числа элементов в неупорядоченной базе данных N (логарифмическая шкала): а) зависимость времени выполнения алгоритмов от числа элементов; б) зависимость памяти, затраченной на выполнение алгоритмов от числа элементов

Практическое сравнение демонстрирует, что при реальной реализации классический алгоритм поиска выполняется быстрее, чем квантовый алгоритм Гровера на языке Q#. Это обусловлено тем, что выполнение алгоритма Гровера осуществляется на классическом симуляторе квантовых вычислений, а также усложнением вычислительного процесса при реализации квантового оракула и управляемых операций.

Кроме того, квадратичное ускорение алгоритма Гровера носит асимптотический характер и проявляется только при достаточно больших размерах пространства поиска при реализации на квантовом компьютере.

Таким образом, результаты эксперимента подчёркивают, что классические алгоритмы поиска остаются эффективными для большинства практических применений при выполнении на классических устройствах, а преимущества алгоритма Гровера проявляются только при работе с реальными квантовыми устройствами.

ЛИТЕРАТУРА

1. Celik, N. Analysis of grover's quantum search algorithm on a classical computer: Identifying opportunities for improvement / N. Celik, O. Bingol // Sigma Journal of Engineering and Natural Sciences. – 2024. – Vol. 42, № 4. – P.1039–1049.

2. Grover, L. K. A fast quantum mechanical algorithm for database search // Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. – 1996. – P. 212-219.

3. Preskill, J. Quantum computing in the NISQ era and beyond // Quantum. – 2018. – Vol. 5. – P. 1–20.

4. Nielsen, M.A. Quantum Computation and Quantum Information / M.A. Nielsen, I.L. Chuang – Cambridge: Cambridge University Press, 2000. – P. 318.

5. Diehl, P. Parallel C++: Efficient and Scalable High-Performance Parallel Programming Using HPX / P. Diehl, S. R. Brandt, H. Kaiser – Springer, 2024. – P. 59–79.