Московский Государственный Университет им. М.В. Ломоносова
Факультет Вычислительной Математики и Кибернетики
Кафедра Суперкомпьютеров и Квантовой Информатики

Практикум на ЭВМ

Отчёт № 1

Параллельная программа на OpenMP, реализующая однокубитное квантовое преобразование

Работу выполнил
Сайбель Т. А.

Москва 2021

Постановка задачи и формат данных

1) Реализовать параллельную программу на C++ с использованием OpenMP, которая выполняет однокубитное квантовое преобразование над вектором состояний длины 2^n, где n – количество кубитов, по указанному номеру кубита k. Для работы с комплексными числами использовать стандартную библиотеку шаблонов.
2) Определить максимальное количество кубитов, для которых возможна работа программы на системе Polus. Выполнить теоретический расчет и проверить его экспериментально.
3) Протестировать программу на системе Polus. В качестве теста использовать преобразование Адамара по номеру кубита:

a) Который соответствует номеру в списке группы плюс 1
b) 1
c) n

Начальное состояние вектора генерируется случайным образом.

Описание алгоритма

Однокубитная операция над комплексным входным вектором $\{\ _i\}$ размерности 2^n задается двумя параметрами: комплексной матрицей $\{\ _{ij}\}$ размера 2x2 и числом от 1 до n (номер кубита, по которому проводится операция). Такая операция преобразует вектор $\{\ _i\}$ в $\{\ _i\}$ размерности 2^n, где все элементы вычисляются по следующей формуле:

$$i_1 i_2 \ldots i_k \ldots i_n = \sum_{j_k=0}^{1} i_k j_k \; i_1 i_2 \ldots j_k \ldots i_n = i_k 0 \; i_1 i_2 \ldots 0_k \ldots i_n + i_k 1 \; i_1 i_2 \ldots 1_k \ldots i_n$$

$$U = \begin{pmatrix} 00 & 01 \\ 10 & 11 \end{pmatrix}$$

Преобразование Адамара задается следующей матрицей:

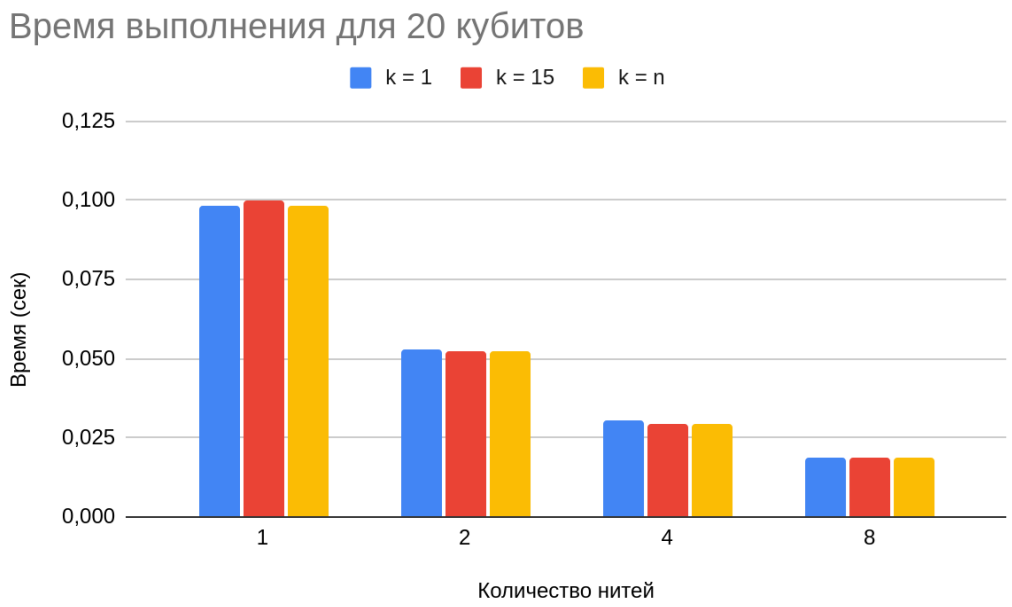
$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

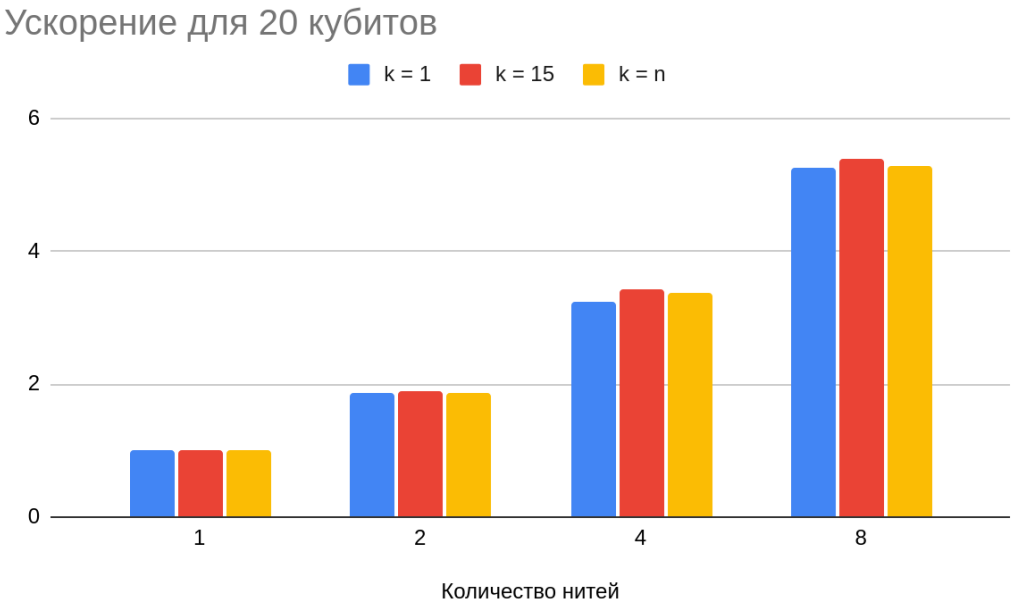
Исследования проводились на вычислительном комплексе IBM Polus.
Для оценки времени выполнения программы использовалась функция omp_get_wtime().
Ускорение, получаемое при использовании параллельного алгоритма для нитей, высчитывалось как отношение времени выполнения программы без распараллеливания к времени параллельного выполнения программы.

Результаты выполнения

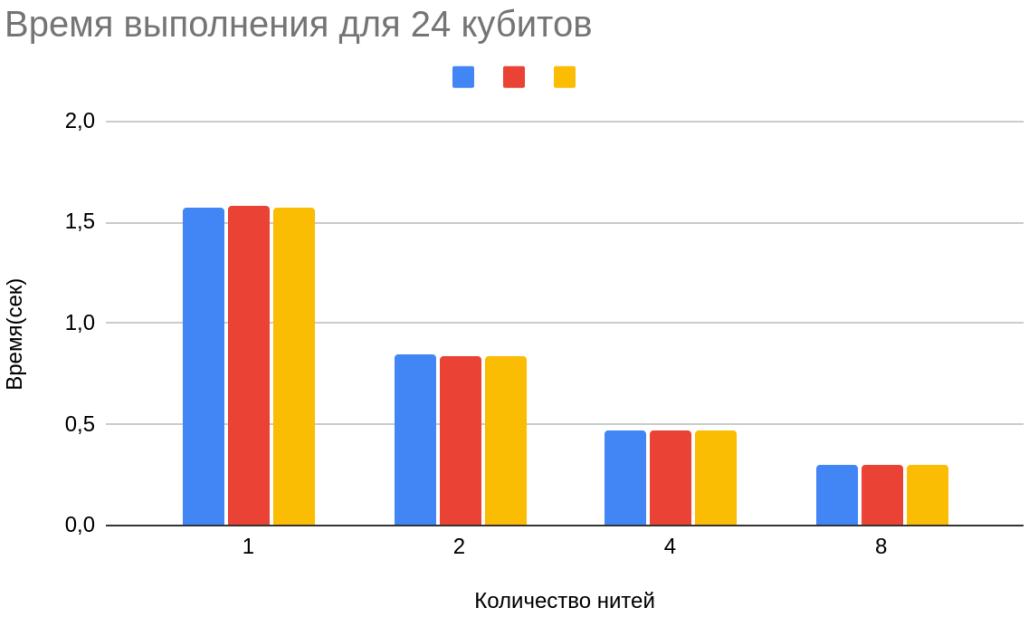
Количество кубитов (n)	Количество нитей	Время работы (сек)			Ускорение		
		k = 1	k = 15	k = n	k = 1	k = 15	k = n
20	1	0,10	0,100232	0,0982357	1	1	1
	2	0,0527058	0,0525821	0,0522565	1,86438115	1,906200019	1,879875231
	4	0,030255	0,0291571	0,0291172	3,247849942	3,437653265	3,373803113
	8	0,0186652	0,0186162	0,0185533	5,264540428	5,384127803	5,294783138
24	1	1,57629	1,57783	1,57583	1	1	1
	2	0,842181	0,840052	0,838593	1,871676041	1,878252775	1,879135647
	4	0,46714	0,469123	0,467594	3,374341739	3,363360995	3,370081738
	8	0,302467	0,299666	0,303805	5,211444554	5,265295362	5,186978489
28	1	25,1737	25,1618	25,1747	1	1	1
	2	13,4646	13,5813	13,6839	1,86962108	1,852679788	1,839731363
	4	7,46355	7,48012	7,48633	3,372885557	3,36382304	3,36275585
	8	4,86818	4,8196	4,75424	5,171070092	5,220723712	5,29521017
33	1	107,448	101,5	101,266	1	1	1
	2	57,117	54,0821	53,7258	1,8811912	1,876776	1,884867
	4	32,9394	31,4278	30,3665	3,2619902	3,229624	3,334793
	8	20,1895	19,1686	19.4004	5,3219742	5,2951	5,219789

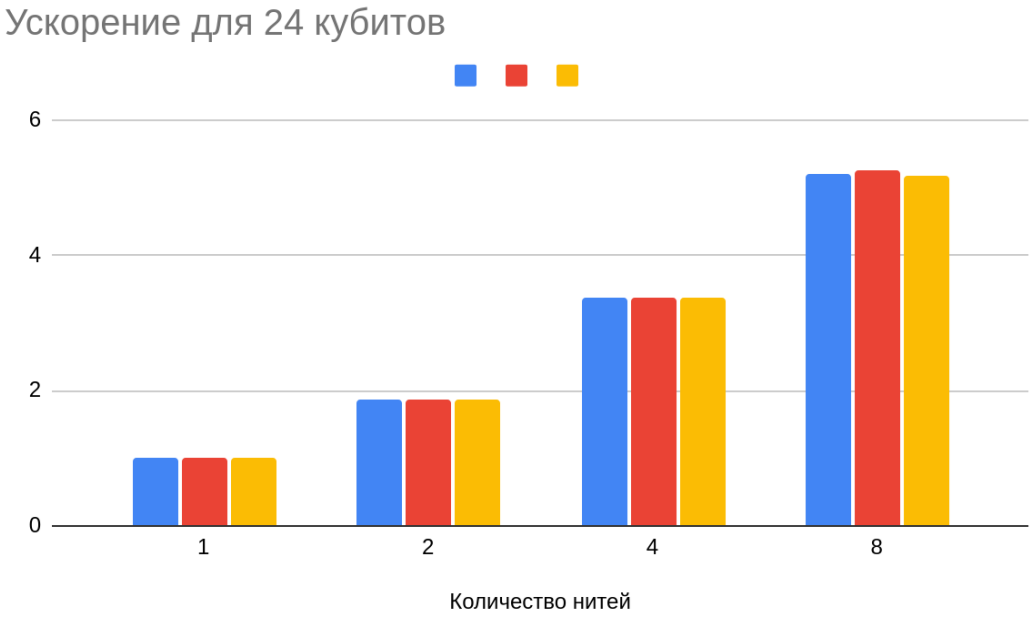
Результаты выполнения для 20 кубитов



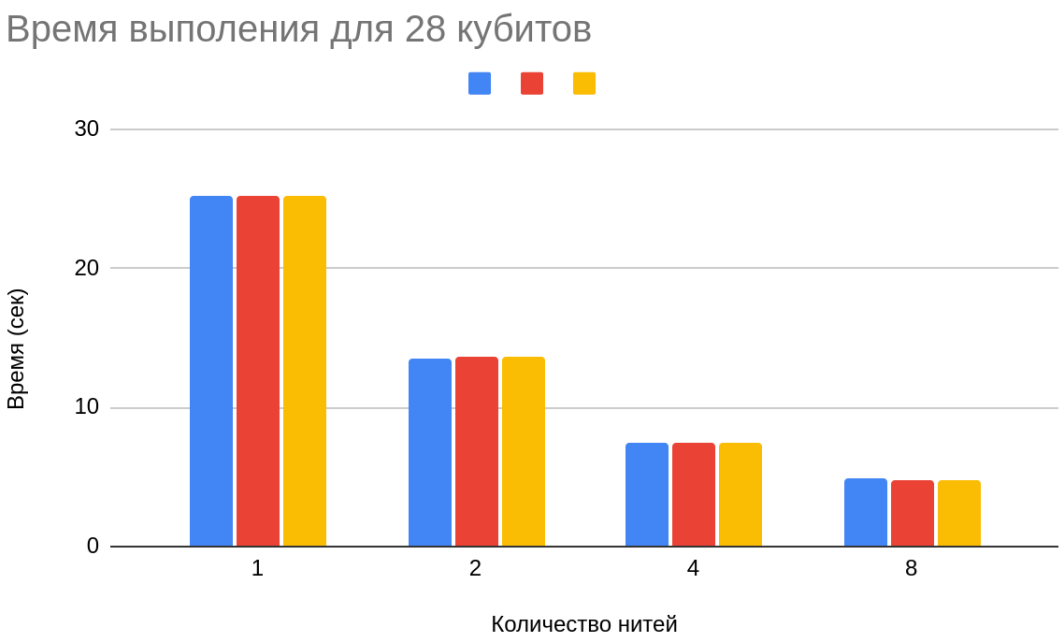


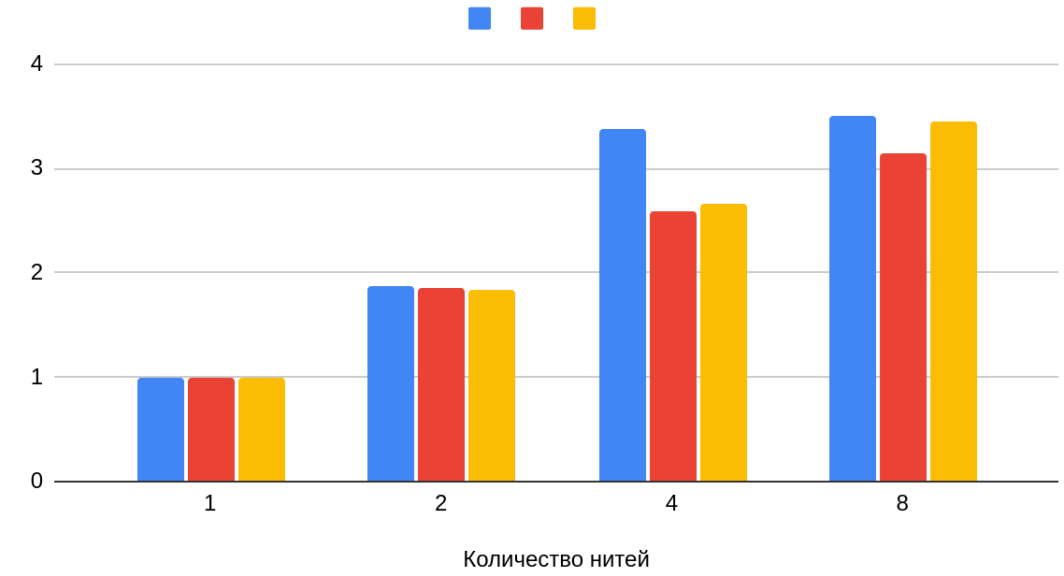
Результаты выполнения для 24 кубитов



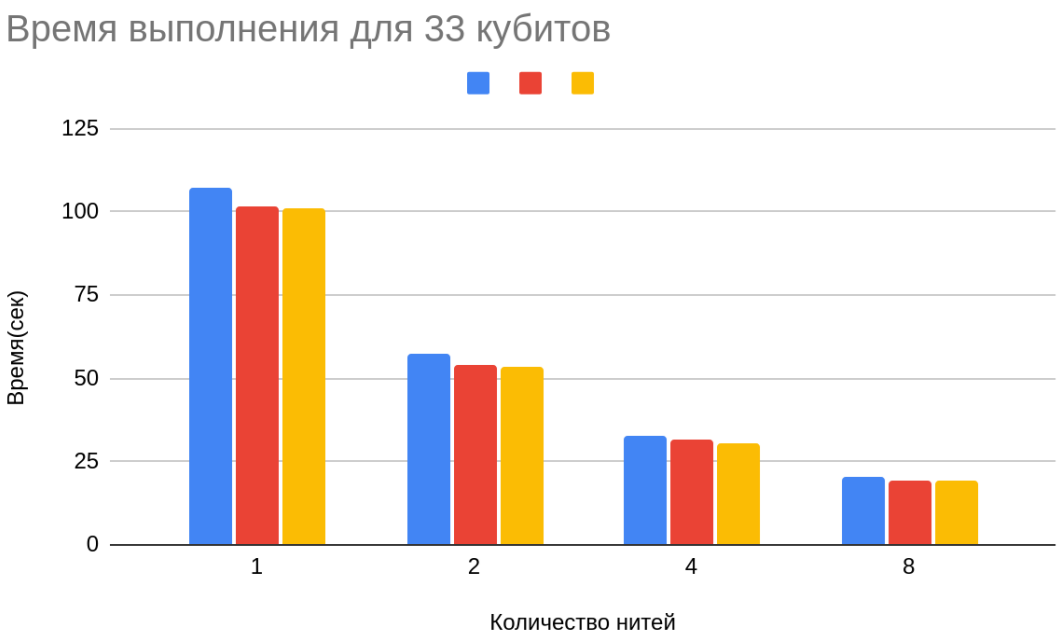


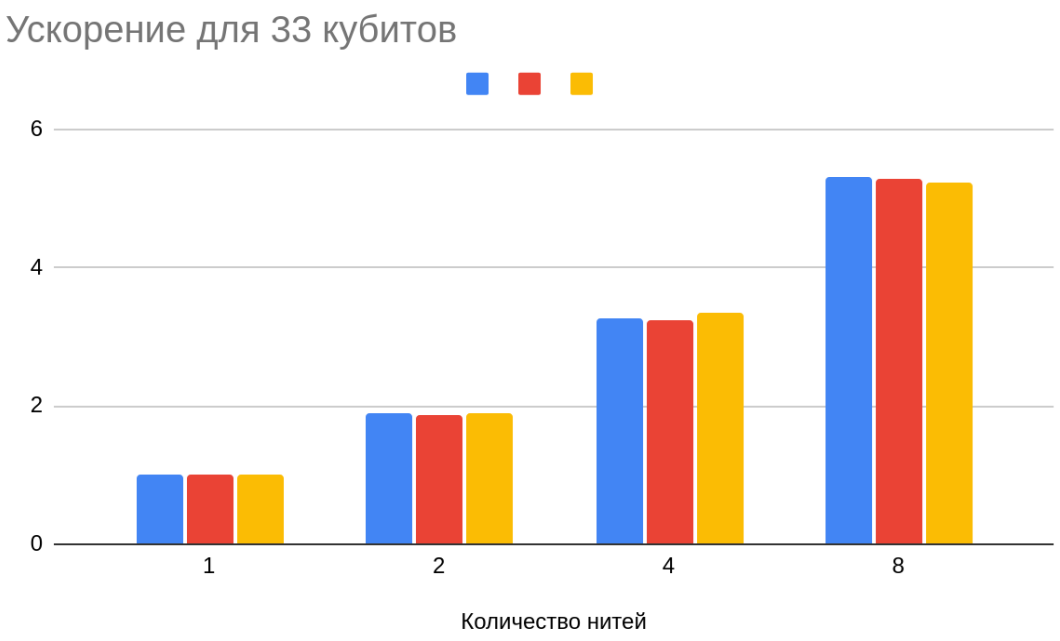
Результаты выполнения для 28 кубитов





Результаты выполнения для 33 кубитов





Основные выводы

При запуске программы с n > 33 возникала ошибка выделения памяти. Поэтому максимальное количество кубитов = 33.

Распараллеливание ускоряет выполнение программы, но с увеличением числа нитей эффективность снижается из-за роста накладных расходов на организацию параллелизма.