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 $\{a_i\}$ размерности 2^n задается двумя параметрами: комплексной матрицей $\{u_{ij}\}$ размера 2x2 и числом k от 1 до n (номер кубита, по которому проводится операция). Такая операция преобразует вектор $\{a_i\}$ в $\{b_i\}$ размерности 2^n, где все элементы вычисляются по следующей формуле:

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

$$U = (uu_{1000}\, uu_{1101})$$

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 p 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	0.0312511	0.0316668	0.0312415	1	1	1
	2	0.0216952	0.0221583	0.0216428	1,4404615	1,429116	1,443505
	4	0.0149701	0.0172135	0.0194342	2,0875678	1,839649	1,607552
	8	0.0137733	0.0155727	0.0148008	2,2689624	2,033481	2,110798
24	1	0.506483	0.507684	0.499817	1	1	1
	2	0.301236	0.301745	0.298605	1,6813495	1,682493	1,673840
	4	0.205312	0.19884	0.201487	2,4668942	2,553228	2,480641
	8	0.169514	0.175549	0.160435	2,9878535	2,891978	3,115386
28	1	8.56725	8.0107	7.97989	1	1	1
	2	4.92314	4.67153	4.65838	1,7402003	1,714791	1,713018
	4	3.21881	3.09776	3.00692	2,6616202	2,585965	2,653841
	8	2.4428	2.55408	2.31516	3,5071434	3,136432	3,446798
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

Основные выводы

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