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

Практикум на ЭВМ

Отчёт № 2

Параллельная программа на MPI, реализующая однокубитное квантовое преобразование

Работу выполнил
Чепурнов А. В.

Москва 2019

Постановка задачи и формат данных

1) Разработать схему распределенного хранения данных и параллельный алгоритм для реализации однокубитного квантового преобразования на кластерной системе.
2) Реализовать параллельную программу на C++ с использованием MPI, 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.
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 и нормируется (тоже параллельно).

Формат командной строки: <номер кубита k> <имя файла исходного вектора> [<имя файла полученного вектора>]
Формат файла-вектора: Вектор представляется в виде бинарного файла следующего формата:

Тип	*Значение*	*Описание*
Число типа int	n – натуральное число	Число кубитов
Массив чисел типа complex<double>	2^n – комплексных чисел	Элементы вектора

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:

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

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

Преобразование Адамара задается следующей матрицей:

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

Аппаратное обеспечение: Исследования проводились на вычислительном комплексе IBM Polus.

Анализ времени выполнения: Для оценки времени выполнения программы использовалась функция MPI_Wtime().

Анализ ускорения: Ускорение, получаемое при использовании параллельного алгоритма для p процессов, высчитывалось как отношение времени выполнения программы без распараллеливания к времени параллельного выполнения программы.

Результаты выполнения

Количество кубитов (n)	Количество процессов	Время работы (сек)			Ускорение		
		k = 1	k = 13	k = n	k = 1	k = 13	k = n
25	1	3,41102	3,40574	3,57164	1	1	1
	2	2,75921	2,37331	1,71909	1,236231	1,435017	2,077634
	4	1,49694	1,02881	1,23041	2,278662	3,310368	2,902805
	8	0,691359	0,536995	0,742303	4,93379	6,342219	4,811566
26	1	9,64961	7,76059	7,69978	1	1	1
	2	5,65844	3,52311	3,51	1,705348	2,202767	2,19367
	4	1,89828	2,49397	2,36554	5,083344	3,111742	3,254978
	8	1,38815	1,71019	1,86202	6,951417	4,537853	4,135176
27	1	13,2173	15,2707	18,2896	1	1	1
	2	7,26341	9,3142	6,80231	1,81971	1,639507	2,688734
	4	3,79213	4,87965	3,40118	3,485455	3,129466	5,377428
	8	2,3723	1,7071	2,49288	5,571513	8,945404	7,336735
28	1	26,1903	26,206	27,2144	1	1	1
	2	14,7563	13,6543	13,6075	1,774855	1,919249	1,999956
	4	7,42059	6,72938	7,96291	3,529409	3,894267	3,417645
	8	5,32408	4,08609	5,59793	4,919216	6,413466	4,861511