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

Отчёт
Однокубитное преобразование вектора-состояния

Работу выполнил
Имашев В.Р.

Москва 2021

Задание

Реализовать параллельную программу на C++ с использованием технологии 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 возможно использование стандартной библиотеки шаблонов.

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

Описание алгоритма

Однокубитная операция задается двумя параметрами: комплексной матрицей размера 2*x*2 и числом от 1 до *n* (данный параметр обозначает номер кубита, по которому проводится операция).

Итак, дана комплексная матрица:

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

и *k* - номер индекса от 1 до *n* (номер кубита).

Такая операция преобразует вектор $\{a_{i_1 i_2 \ldots i_n}\}$ в $\{b_{i_1 i_2 \ldots i_n}\}$, где все 2^n элементов нового вектора вычисляются по следующей формуле:

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

Реализация алгоритма с помощью MPI

На каждом процессе хранится лишь фрагмент вектора длиной исходный вектор / количество процессов. При выполнении преобразования вычисляется номер процесса, на котором находится инвертированный бит с номеров к.

В случае, если номер процесса совпадает с текущим, преобразование производится по обычной схеме в соответствии с формулой выше. Иначе происходит обмен данными между текущим и процессом, в котором находится инвертированный бит и далее преобразование производится также по обычному сценарию в соответствии с формулой, однако с той лишь поправкой, что в буфере, куда производится запись нового вектора, уже лежат все необходимые инвертированные биты.

Тестирование на Polus

Тестирование программы проводилось на Polus. На следующей странице приведена таблица, содержащая информацию о результатах выполнения программы на вычислительном комплексе.

Замер времени производился с помощью функции MPI_Wtime().

Количество кубитов	Количество процессоров	Время работы программы(сек)			Ускорение		
		k =1	k = 8	k = n	k = 1	k = 8	k = n
20	1	0,0948683	0,0949739	0,0903856	1	1	1
	2	0,0463643	0,0455327	0,0474659	2,04615	2,085839	1,904222
	4	0,0374606	0,0229512	0,0234135	2,532482	4,13808	3,860405
	8	0,0116401	0,0115699	0,0157496	8,150128	8,208705	5,738914
	16	0,00835781	0,00846789	0,00899738	11,35086	11,21577	10,04577
	32	0,00744425	0,00769615	0,00805434	12,74384	12,34044	11,22197
	64	0,0050343	0,00425493	0,00484385	18,84439	22,32091	18,65987
24	1	1,54115	1,46428	1,50461	1	1	1
	2	0,797478	0,726039	0,76084	1,93253	2,016806	1,977564
	4	0,473435	0,370341	0,347588	3,255252	3,95387	4,328717
	8	0,209681	0,167012	0,213794	7,349974	8,767514	7,037662
	16	0,136731	0,118364	0,13612	11,2714	12,37099	11,05356
	32	0,0787681	0,0474588	0,0712384	19,56566	30,85371	21,12077
	64	0,046337	0,0362297	0,029303	33,2596	40,41656	51,34662
28	1	21,2542	22,5342	23,3167	1	1	1
	2	12,7003	11,7403	11,9609	1,67352	1,919389	1,94941
	4	5,96749	5,82549	5,94698	3,561665	3,868207	3,920763
	8	2,83873	2,96873	3,05866	7,487221	7,590519	7,623175
	16	1,45436	1,53338	1,62246	14,61413	14,69577	14,3712
	32	0,896484	0,928583	1,03373	23,7084	24,2673	22,55589
	64	0,374153	0,421283	0,635265	56,80617	53,48946	36,7039
максимальное (удалось при 30)	1	-	-	-	-	-	-
	2	-	-	-	-	-	-
	4	-	-	-	-	-	-
	8	12,4221	11,7204	12,1317	1	1	1
	16	5,89532	6,19438	6,04367	2,107112	1,892102	2,00734
	32	2,89628	3,37418	3,23035	4,288984	3,473555	3,755537
	64	1,86345	1,96765	2,06713	6,666184	5,956547	5,868862

Выводы

Уменьшение скорости падения времени работы программы с ростом числа процессов (иными словами, уменьшение ускорения работы программы) связано с возрастанием количества межпроцессорных обменов (в программе используется, например, обмены с помощью функции MPI_Sendrecv), которые занимают большую часть времени работы.

При небольшом увеличении числа кубитов время программы растет заметным образом (например, при увеличении числа кубитов на 4, время работы увеличивается в среднем почти в два раза).