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

Курс: Практикум
Параллельная программа на OpenMP, которая реализует однокубитное квантовое преобразование.

Работу выполнил
Мокров К.С.
323 группа

Москва 2018

Задание

1.Реализовать параллельную программу на C++ с использованиемOpenMP, которая выполняет однокубитное квантовое преобразование над вектором состояний длины 2n, где n–количество кубитов, по указанному номеру кубита k. Описание однокубитногопреобразования дано ниже в разделе методические рекомендации[1]. Для работы с комплексными числами возможно использование стандартной библиотеки шаблонов[2].

2.Определить максимальное количество кубитов, для которых возможна работа программы на системе Regatta. Выполнить теоретический расчет ипроверить егоэкспериментально.

3.Протестировать программу на системе Regatta. В качестве теста использовать преобразование Адамарапо номеру кубита: a)который соответствует Вашему номеру в списке группы плюс 1. b) 1 c)n

Начальное состояние вектора должно генерироваться случайным образом. Заполнить таблицу и построить график зависимости ускорения параллельной программы от числа процессоров для каждого из случаев a)-c)

4.Написать отчет, который будет содержать листинг программы, а так же результаты выполнения пунктов 2-3.

Результат

		Время работы программы			Ускорение		
		Номер кубита					
Количество кубитов	Количество нитей	7	1	n	7	1	n
20	1	0,0918983	0,0995914	0,0925905	1	1	1
	2	0,061798	0,0515748	0,0411891	1,487075634	1,931008942	2,247936954
	4	0,0356126	0,0417705	0,0388404	2,580499598	2,384252044	2,383870918
	8	0,0432656	0,0288844	0,0181169	2,124050054	3,447930371	5,110725345
24	1	1,31083	1,3218	1,13467	1	1	1
	2	0,749072	0,820327	0,695232	1,749938591	1,611308661	1,632073898
	4	0,441631	0,40793	0,512544	2,968156674	3,24026181	2,213800181
	8	0,3289	0,198693	0,372985	3,985497112	6,652473917	3,042133062
28	1	23,1217	21,2854	22,3742	1	1	1
	2	11,0415	11,0791	12,7471	2,094072363	1,921221038	1,755238446
	4	5,04724	6,02767	6,03436	4,581058162	3,531281573	3,707799999
	8	5,35367	3,79349	5,35577	4,318850433	5,611033639	4,17758791
30	1	88,5506	92,7134	86,7016	1	1	1
	2	50,0464	48,7432	49,2589	1,769370025	1,902078649	1,760120506
	4	29,4703	36,0957	22,7189	3,004740366	2,568544176	3,816276316
	8	19,5321	19,358	18,7031	4,533593418	4,789410063	4,635680716

Изначально планировалось взять вектор размера 2^32, так как 256Гбайт = 2 вектора по 2^n элементов размера 16, получаем n = 33, плюс накладные расходы, итого n = 32, но ,видимо, нам не дают всю доступную память, поэтому пришлось взять наибольшее число, с которым программа работала.

Как видно программа действительно начинает работать быстрее при увеличении

количества потоков, но если посчитать эффективность(Эффективность = Ускорение / количество потоков), то мы нигде не получив числа большего единицы, что говорит нам, о том что программа не очень хорошо параллелизуется.