Московский государственный университет имени
М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра системного анализа

Отчёт по практикуму

«Суперкомпьютеры и параллельная обработка данных»

Параллельная реализация алгоритма быстрой сортировки

Студент 415 группы
А. Н. Ашабоков

Москва, 2019

Содержание

1 Обоснование выбора алгоритма для параллельной реализации

Алгоритмы сортировки являются одними из наиболее важных и часто используемых алгоритмов при написании программ, связанных с обработкой больших массивов данных, а алгоритм быстрой сортировки является наиболее эффективным и распространенным.

2 Ссылка на описание выбранного математического алгоритма в энциклопедии AlgoWiki

Описание выбранного алгоритма можно найти по ссылке:
algowiki-project.org/ru/Участник:Ashabokov_415/Алгоритм_быстрой_сортировки

3 Результаты запусков на суперкомпьютере Ломоносов

Для оценки эффективности и масштабируемости программы были проведены серии испытаний с различными конфигурациями. Для каждой конфигурации (N, P), где N — длина входного массива, P — число задействованных процессоров, было произведено по 3 запуска программы, после чего производилось усреднение полученных для фиксированной конфигурации результатов. В качестве входных данных были взяты случайным образом сгенерированные массивы длины $[500 : 2300]$ с шагом 200, значения количества задействованных процессоров были взяты из диапазона $[8, 128]$ с шагом 8. Результаты испытаний приведены в таблице:

	N																
		8	16	24	32	40	48	56	64	72	80	88	96	104	112	120	128
	500	0	0.035	0.03	0.06	0.055	0.055	0.055	0.08	0.075	0.075	0.08	0.07	0.08	0.085	0.07	0.1
	700	0	0.045	0.04	0.065	0.07	0.065	0.07	0.095	0.09	0.085	0.09	0.09	0.085	0.09	0.075	0.105
	900	0	0.025	0.03	0.045	0.055	0.055	0.055	0.065	0.075	0.06	0.07	0.06	0.08	0.065	0.095	0.075
P	1100	0	0.035	0.045	0.07	0.065	0.065	0.07	0.095	0.095	0.09	0.09	0.09	0.095	0.095	0.08	1.585
	1300	0	0.04	0.045	0.07	0.065	0.065	0.07	0.09	0.09	0.085	0.09	0.1	0.09	0.09	0.085	0.105
	1500	0	0.04	0.4	0.65	0.06	0.06	0.06	0.085	0.09	0.085	0.085	0.085	0.085	0.08	0.095	0.11
	1700	0	0.04	0.45	0.06	0.055	0.065	0.065	0.085	0.08	0.08	0.095	0.085	0.08	0.085	0.09	0.105
	1900	0	0.035	0.035	0.06	0.065	0.06	0.065	0.085	0.095	0.09	0.1	0.085	0.08	0.085	0.085	0.1
	2100	0	0.045	0.04	0.07	0.07	0.075	0.07	0.09	0.1	0.09	0.08	0.095	0.095	0.1	0.1	0.115
	2300	0	0.03	0.03	0.065	0.05	0.055	0.05	0.075	0.08	0.08	0.9	0.08	0.06	0.07	0.085	0.1

Таблица 1: Время выполнения программы для различных конфигураций.

4 График сильной масштабируемости с пояснениями

Для начала введем необходимые термины.

Определение 1 *Сильная масштабируемость (strong scaling) - зависимость производительности R от количества процессоров p при фиксированной вычислительной сложности задачи ($W = const$).*

Определение 2 *Производительность (performance) $R = Op/t$ определяется количеством операций Op, производимых данным компьютером в единицу времени.*

Определение 3 *Вычислительная сложность (computational complexity) задачи W - количество основных вычислительных шагов лучшего последовательного алгоритма, необходимых для решения задачи на одном процессоре.*

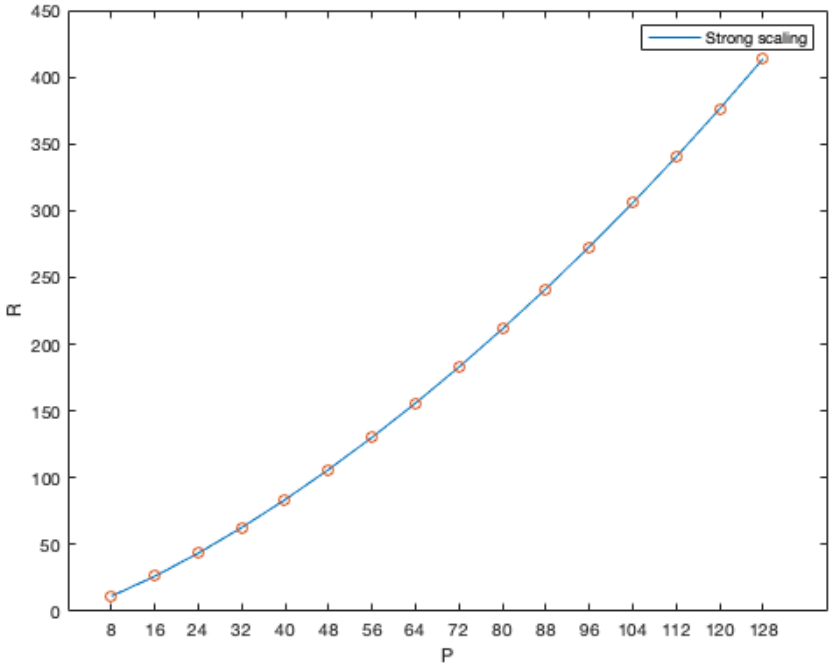

Рис. 1: График сильной масштабируемости.

Из приведенного выше графика видно, что с ростом количества процессоров P происходит рост производительности R при фиксированном значении вычислительной сложности задачи W.

5 Объяснение полученных результатов

На Рис.1 и приведенных в статье [2] графиках можно наблюдать рост производительности при увеличении числа процессоров. Относительно быстрый рост производительности, наблюдаемый на Рис.1 можно объяснить тем, что при увеличении числа процессоров происходит сильное сокращение числа сортируемых на каждом из процессоров данных. Из приведенных в [2] рассуждений следует, что данный алгоритм имеет относительно хорошую положительную масштабируемость по числу процессоров и отрицательную масштабируемость по числу входных данных. Правда, стоит отметить, что скорость убывания эффективности при увеличении числа входных данных крайне мала, что приводит к выводу, что показатель масштабируемости по двум направлениям положителен, и при увеличении числа процессоров и числа входных данных наблюдается рост эффективности.

6 Сведения о программно-аппаратной среде

Все вычисления производились на суперкомпьютере "Ломоносов". В настоящее время он содержит 6654 вычислительных узла, более 94000 процессорных ядер, обладает пиковой производительностью 1,37 Пфлоп/с. Реальная производительность системы на тесте Linpack равна 674 Тфлоп/с, что позволило ему занять в июне 2011 года 13–ое место в списке Top500 самых мощных компьютеров мира. [3]

Программа написана на языке Си. Для компиляции использовались компиляторы openmpi/1.8.4-gcc и gcc/5.5.0. Версия slurm: slurm/2.5.6.

Компиляция:

```
mpicc ~/main.c -o ~/_scratch/main
```

Запуск:

```
sbatch -p test -n P ompi ~/_scratch/main ~/_scratch/input_file_name
```

где $P = 2^N$ — количество процессоров, input_file_name — имя текстового файла, содержащего массив из целых чисел, записанных через пробел.

7 Библиография

Список литературы

[1] Воеводин Вл.В. *Лекции по курсу „Суперкомпьютеры и параллельная обработка данных“.*

[2] algowiki-project.org/ru/Участник:Ashabokov_415/Алгоритм_быстрой_сортировки — Открытая энциклопедия свойств алгоритмов.

[3] www.msu.ru/lomonosov/science/computer.html — Суперкомпьютер «Ломоносов».